PERSONALNI
računari
BIGZ 1971
94
SEPTEMBAR 1993.
CENA 1500000000 D.
DEM 10 ○ USD 6 ○ ATS 70

NAJVEĆA NAGRADNA IGRA U ISTORIJI ČASOPISA RAČUNARI

45.000 DEM U HARDVERU I SOFTVERU

ŠTAMPAČI
HP LASERJET 4L
AEG OLYMPIA 60P
STAR ZA-250

OPERATIVNI SISTEMI
KOJI VAM JE WINDOWS POTREBAN?

GENERISANJE ZVUKA
ZVUČNE KARTICE

SOFTVER
ARJ 2.41
BODY ATLAS
CORELDRAW 4.0

EKSKLUZIVNA NAGRADA
LASERSKI ŠTAMPAČ HP L4 FIRME COMPUTER IZ BEOGRADA

DRUGA NAGRADA
PC PLUS
386/40 MHz

SPECIJALNI DODATAK NA 32 STRANE
MOJA ŠKOLA PASKALA

CAD RAČUNARI

CAD RAČUNARI I OPREMA

CAD računari i oprema
386 i 486 računari optimizovani za CAD i grafičke namene
DIAMOND i HERCULES procesorske grafičke kartice
EIZO kolor monitori visoke rezolucije
HEWLETT-PACKARD kolor štampači, skeneri i ploteri
POLAROID film rikorderi

SOFTWARE

AutoCAD r.12
AutoSHADE
ANIMATOR Pro
3D Studio 2

USLUGE

AutoCAD škola
CorelDRAW škola
RENTIRANJE plotera, laserskih i kolor štampača
kolor skenera,
film rikordera

KONSALTING

ArhiCAD i Arhi 24
globalna rešenja za primenu CAD-a u arhitekturi

CAD RAČUNARI STVARAJU UMETNOST

PREDUZEĆE
MICRO ANIMA
Beograd, 29 novembra 71
Tel 343 940 / 343 294

MICRO

ANIMA

PERSONALNI
računari 94

časopis za informatiku i računarstvo
YU ISSN 0352-7221
Izdaje BIGZ
septembar 1993

Izdaje i štampa
Beogradski zdavačko-grafički zavod
11000 Beograd
Bulevar Vojvode Mišića 17
Generalni direktor
Ilija Rapajć
Glavni i odgovorni urednik
Jovan Regasek
Urednik
Ranko Jovanović
Komercijalni urednik
Vesna Jeremić
Tehnički urednik
Redoje Kavedžić
Stručna redakcija
Nenad Batočanin (baze podataka)
Renka Jovanović (aktuelnosti) Dejan Ristanović (softver) Jovan Skuljan (programiranje) Vladimir Stamenović (radne tabele) Dejan Vasiljević (hardver) Dragutin Vuković (mreže) Zoran Životić (softver)
Adresa redakcije
11000 Beograd
Bulevar vojvode Mišića 17/III
Telefoni
647-476 (glavni urednik)
653-748 (redakcija)
851-866 (centrala pretplata 226 stari brojevi 259)
Agencija BIGZ-a
(011) 651-733 653-565
Telefaks
(011) 648-140 647-955
Pretplata na zemlju

Za 3 meseca	zvati prodaju
Za 6 meseci	zvati prodaju
Za 11 meseci	zvati prodaju

(na žiro-račun: D P BIGZ, 60802-603-23264)
Pretplata na inostranstvu
Za jednu godinu 51 USD 80 DEM 67 CHS 27 GBP 271 FRF
(na devizni račun D P BIGZ 60811-620-16101-820701-999-03377)
Zbog usporenog prenošenja uplata preko banke molimo pretplatnike da nam pošalju kopiju uplatnice
Rukopisi se ne vraćaju

SEZAM (011) 648-422 (16 linija)
sistem za modemske komunikacije
Upravnik sistema
Jovan Regasek
Administratori na sistemu
Zoran Životić i Dejan Ristanović

CENOVNIK OGLASNOG PROSTORA

Cene oglasnog prostora ta izražavaju u 'bodovima' čija vrednost odgovara vrednosti nemačke marke na slobodnom tržištu na dan fakturisanja

1/1 poslednja strana korica	1 800 bodova
1/1 druga strana korica	900 bodova
1/1 treća strana korica	900 bodova
1/1 kolor unutrašnje strane	900 bodova
1/2 kolor unutrašnje strane	500 bodova
1/1 crno-bela strana	550 bodova
1/2 crno-bela strana	400 bodova
1/4 crno-bela strana	300 bodova
1/8 crno-bela strana	200 bodova
mali oglas (po centimetru)	20 bodova

Fakturisanje se vrši na dan ugovaranja oglasnog prostora prijema potrebnih materijala za oglas sa obavezom da se uplata izvrši pre izlaska broja iz štampe

Rok za dogovor i rezervaciju oglasnog prostora je 35 dana pre izlaska broja iz štampe

Molimo vas da se za dogovore i informacije javite na telefon 011/653-748 fax 011/648-140 modem 011/648-899 (mail write redakcija) Kontakt: Vesna Jeremić

SADRŽAJ

4 Vesti
ŠTA IMA NOVO
10 Laserski štampači / HP 4L
TEMPERAMENTNI ŠTEDIŠA
12 Laserski štampači / AEG LP60
SAD ILI NIKAD
15 Arhitektura računara / Pentium ploče
PENTIUM IZ TEMELJA MENJA PC SVET
19 Operativni sistemi / Familija Windows
BILO KUDA WINDOWS SVUDA
29 Matrični štampači / Star ZA-250
ZVEZDA U OFISU
32 Nagradna igra
KATALOG NAGRADA
36 Arhiveri / ARJ 2.41
DOK SU DRUGI SPAVALI
40 Medicina / BodyAtlas
PUTOVANJE KROZ LJUDSKO TELO
43 Softverske alatke / Formatiranje disketa
DO POSLEDNJEG BAJTA
46 Grafički softver / CorelDraw 4.0.
NIKO KAO COREL
52 Zvučne kartice
PC ONOMATOPEJA
56 Mikroprocesori / i486
VIŠEPROGRAMSKI RAD
58 Programeri govore / Struktura DOS-a
ŠESTICA IZNUTRA
62 Softverske tehnike / Keširanje diska
INTELIGENTNI KONTROLERI
64 Softverske tehnike / Pretraživanje teksta
OA LI JE NEKO VIDEO TANJU?
66 Objektno programiranje
SUDAR SVETOVA
68 Tehnike programiranja / Clipper
OBJEKTI ZA NESTRPLJIVE
72 Tehnike programiranja / Paradox Engine
ZAMENITE MOTOR AKO SU VAM PODACI SPORI
76 Računanje vremena
STROGO PO KALENDARU
80 Rezidentni programi
TSR UZVRAĆA UDARAC
84 Dejanove pitalice
ŠAHOVSKA PITALICA
87 Dejan Ristanović
BAJTOVI LIČNE PRIRODE
90 Dejan Ristanović
MOJA ŠKOLA PASKALA
91 Zoran Životić
MOJA ŠKOLA C-a
92 Nenad Batočanin
MOJA ŠKOLA CLIPPER-a
94 Pavle Pekovič
MOJA ŠKOLA UNIX-a
94 Biltени
SEZAM BILTEN
SEZAM FILE

Lepa čelistkinja na našoj naslovnoj strani je Snežana Pešić, maturant gimnazije iz Niša. Na njenom koncertu, plemenite zvuke vielen-cela, umesto simfonijskog orkestra, prati muzika iz PC računara - koja zvučna kartica može da na nosi na taks virtuoznom ukegom, pročitajte u tekstu Aleksandra Šušnjara "PC onomatopeja" na str. 52-56. Koncert je, specijalna za fotografsku kameru radionica Studio Tešić, priređen na sceni Jugoslovenskog dramskog pozorišta, a na atmosferu na pobrinuo Milo Bulurević-Kum, dizajner sveta. U kadar smo ugurali i štampač Hewlett Packard 4L, koji predstavlja apsolutni hit sezone. Detalje potražite u prikazu na stranama 10-13, a uz malo sreće jedan primerak se, zahvaljujući firmi **COMPUTER** iz Beograda koja na nedavnu uključila u našu nagradnu igru, može preseliti i u – vašu kuću!

SADRŽAJ OGLASA

ADA COMPUTER	42
ADACOM	7
BS PROCESOR	18
CIM	75
COMTEC	22
CONTRAD	4K, 24
INFO-D	48
INSTITUT B.KIDRIČ	79,86
INTERSOFT	75
JUGODATA	9
MICRO AIR	2K
MICROSYS	20
MLACAR & CO	40
MP BIRO	3K
MZ	85
NOVOSADSKI SAJAM	86
OLIVETTI	30, 31
OLYMP	61
OSA	12
PC PLUS	26
RADIO INDEX	50
RADIO PINGVIN	48
SAGA	28
SPRINT	75
ŠUTLIĆ	51

NAGRADNA IGRA
45.000 DEM ZA ČITAOCE

Zaštita podataka

Klintonova vizija digitalizovane Amerike

Klintonova administracija ima svoju viziju o digitalizovanoj Americi, a sastavni deo te vizije je i čip za brzo šifrovanje pruka, koji treba da pomogne kompanijama i pojedincima da zaštite avoje tajna od znatiželjnika pri prenošenju poruka komunikacionim kablovima.

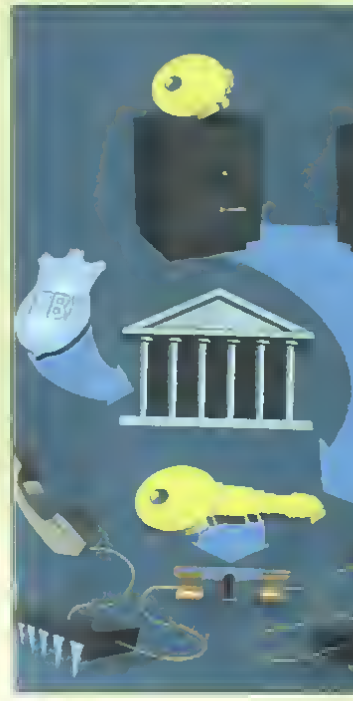
Ali, kvaka je u tome što će čip imati posebne „vratance" kroz koja će ovlašćene državne ustanove ipak moći da pnsluškuju Bele kuća veruje da će ovakav hardver štititi pravo na privatnost svakog Amerikanca, pružajući mu istovremeno i zaštitu od onih koji krše zakon

Čip je nazvan *Clipper* (firma „Intergraph" iz Atabame proizvodi procesor koji nosi isto ime, pa će Clipper-ov moniker verovatno biti izmenjen). To je 12-Mbps koprocesor projektovan u „Mykotronix" u, a proizvodi ga VLSI Smešten je u sigurnom omotaču obezbeđenom od „provale" što će osujetiti napore onome ko bude poželeo da otkrije klasifikovane algoritme koji se nalaze unutre

Osim činjenice da omogućava vladi da ugrozi privatnost korisnika, još jedna smetnja stoji na putu širem prihvatanju *Clipper*-a – njegova cena Ben Stolz iz kompanije „Sun Microsystems" ističe da, „po previlu deo koji košta *n* dolara povećava ukupnu cenu računara za *3n* ili *4n* dolara" Šef američkog Nacionalnog instituta za standarde i tehnologiju je nedavno izjavio da bi „Mykotronix"-ov čip mogao koštati 26 dolara, što bi značilo potencijalno poskupljenje računara za 75 do 100 dolara.

Kritičari *Clipper*-a ističu da ni jeftiniji čipovi koji nude DES šifrovanje nisu stekli širu popularnost, jer softversko šifrovanje, mada obično sporije od hardverskog nije toliko skupo Kompanija „RSA Data Security" čije će kriptografske rutine biti ugrađene u nove verzije sistemskog softvera koje pripremaju „Apple" i „Novell", a *Lotus Notes* ih već koristi smatra da je *Clipper* samo još jedna otrovna strela uperena direktno protiv njihovog proizvoda

Međutim, sama američka vlada će već u startu predstavljati veliko tržište za novi čip Predsednik Klinton je već naložio državnom tužiocu da kupi nekoliko hiljada komada za viedine računare i telefone No uspeh *Clipper*-a u ostalom delu sveta će biti upravljan isključivo ekonomskim razlozima



Preme *Clipper* scenariju, policija ima zakonsko pravo da prisluškuje te da posebnim ključem otključava "trezor" u kojem je pohranjen serijski broj čipa koji je proizveo poruku Vlada, teoratski, dobija zakonsko ovlašćenje (tj sudski nalog), te agencijama zaduženim za čuvanje ključeva može da predoči uredna dokumentaciju. Nakon toga, uz pomoć dva ključa dešifruje poruku.

ŠTA IMA NOVO

TRŽIŠTE

Windows NT

Gejtsova obećanja: Kad ga vidiš, nadaj se

Na izložbi PC Expo koja je krajem juna održana u Njujorku Bil Gejts je održao konferenciju za štampu i između ostalog dao izvesna obećanja – da će *Windows NT* izaći do 24 jula da se novi Microsoft-ov operativni sistem Chicago može očekivati za oko godinu dana a prve aplikacije koje koriste OLE 2 0 tokom tročeg kvartala ove godine

U katkom izveštaju koji je izneo tom prilikom, Gejts je najavio i da će *Microsoft Excel 5.0* i *Microsoft Project 4.0* nova izdanja koja se očekuju ove jeseni sadržati sistem za programiranje iz *Visual Basic*-a te kasnvni *Applications Edition* makro jezik sa podr škom za *OLE Automation 2.0*

Jedan novinar je uhvatio Gejtsa za reč – primetio je da predsednik Microsoft-a u svom izveštaju pomenuo da će se NT pojaviti „u roku od 60 dana" i dodao „To ste rekli i na poslednjem Comdex-u " Gejts je odgovorio „Kad sam to rekao? 24 maja? Pa, dobro onda Windows NT izlazi 24 juta."

Prodaja

Windows aplikacije idu kao alva

Udruženje izdavača softvera (SPA – *Software Publishers Association*) je nedavno iznelo podatak da je u Severnoj Americi prodaja programa za *Microsoft Windows* prvi put premašile prodaju aplikacija pisanih za MS-DOS i to u sve četiri glavne poslovne kategorije tekst-procesori radne tabele baze podataka i grafičke prezentacije U prvom kvartalu ove godina prodaje *Windows* programa je porasla za 114% u odnosu na prethodni deo je prodaja DOS aplikacija opa la za 20%

Očigledno je da sve više korisnika prelazi na *Windows* koji se pojavio još sredinom osamdesetih ali je popularnost stekao tek posle pojave verzije 3 0 u maju 1990 Tranutno se po rečima Microsofta, mesečno proda prako milion kopija *Windows*-a

CD ROM na Marsu

Izvinite, jeste li vi pali sa Zemlje?

Ako na Marsu ima Marsovaca onda bi jednog dana mogli da saznaju kako su ih sve Zemljani zamišlja li, i to zahvaljujući CD ROM projektu na kojem zajednički rade SAD i Rusija.

Američko Planetarno društvo (Planetary Society) u saradnji sa Ruskim institutom za istraživanje svemi na (IKI) pripreme CD-ROM pod nazivom *Visions of Mars* koji na crvenu planetu treba da otputuje iz Ru sije sradinom sledeće godine istelicom Mars 94 *Visions of Mars* će biti zbirka naučno fantastičnih priča, zvučnih zapisa i slika na kompakt-disku svojevrsna hronika ljudske fascinacije Marsom i njegovim zamišljenim stanovnicima počevši od romana engleskog književnike H D2 Velsa (H G Wells) pa sve do današnjih dana Po jedna kopija diska će biti ostavljena u dva male stanice u koja će se Mara 94 spustiti u Septembru 1995

Na žalost stanice neće imati CD ROM drajv, take da će on za sada imati značaj samo ze nas – kao kapsule u kojoj je sačuvana istorija „kao motivacija i memento za buduće istraživača Marsa" kako kaže čuveni doktor Karl Segan (Carl Sagan) predsednik Planetarnog društva

Proizvodič je „Time Warner Interactive Group" iz Kalifornije a još se ne zna da li će se CD-ROM inače predviđen i za Apple i za IBM računare prodavati na tržištu ili će se redit ograničen broj samo za ovu priliku Bilo kako bilo primerci koji će se spustiti na Mara nosiće objašnjenja na pet jezika o *tome* ste je unutra i kako se pokreće „Mikro-lačkica" (*microdot*) na površini diska će sadržati dodatne tehničke podatke

Još uvek se primaju prilozi iz zemalja van engleskog govornog područja a sve će biti zabeleženo na izvornom jeziku Eto eko naši izdavači nemaju sluha za vaše marsovske priče javite se na adresu *Planetary Society Susan Lendroth 818-793-5100.*

HARDVERU JE POTREBAN SOFTVER

VAMA SU POTREBNI *računari*

BEOGRAD
Kuća knjižarstva
„Kultura"
Terazije 45, 011/338-985, 330-627

BEOGRAD
„Kultura"
Bulevar revolucije 289 011/418-731

BEOGRAD
„Kultura"
Kosovska 37 011/321-250

BEOGRAD
„Kultura"
Pešačka 61 011/553-361

BEOGRAD
„Kultura"
Ratka Vojvoda Čeče 28 011/5336-436

BEOGRAD
Patrice Lumumba 46 011/771-094

BEOGRAD
Pešačka 136 011/558-226

NOVI BEOGRAD
Narodnih heroja 11 011/601-522

ALEKSINAC
Maršala Tita 133 018/873-018

BABUŠNICA
27. jula 3 010/ 85-006

BANJA LUKA
Maršala Tita 67-a 038/ 28-912

BANJA LUKA
Maršala Tita 67-a 078/ 26-633

BAR
Jovana Tomaševića 13 27-291 085/ 27-299

BELA CRKVA
Prvog oktobra 78 013/851-000

BOR
Radulfka 4 085/ 31-240

DRVAR
Maršala Tita 168 077/622-576

JAGODINA
Knegiaja Miloša 14 035/222-775

KNJAŽEVAC
Trg oslobođenja 10 019/ 41-238

KULA
Maršala Tita 250 025/722-442

LAZAREVAC
XIII proletarske 16 011/8124-625

JEFTINO
U KNJIŽARAMA BIGZ-a
POPUST 20 %

JEFTINIJE
U PRODAVNICI BIGZ-a
POPUST 30%

Prodavnica BIGZ-a, Bulevar vojvode Mišića 17, 11000 Beograd, je otvorena svakog radnog dana od 7 – 15 sati. Dođite ili rezervišite svoj primerak na (011) 651-666, lokal 259 ili 226.

NAJJEFTINIJE
U PRETPLATI

PRETPLATA

Pretplata je najjeftiniji i najsigurniji način da dođete do svog primerka lista. **Najjeftiniji** zato što je danas sve jeftinije nego što će biti sutra i zato što pretplatnicima odobravamo posebne popuste. **Najsigurniji** zato što ne treba da razmišljate da li ćete „Računare" naći u prodaji ili ne. Nudimo Vam dve mogućnosti:

- DOSTAVU PREPORUČENOM POŠILJKOM
- LIČNO PREUZIMANJE U PRODAVNICI BIGZ-a

Važi prodajna cena na dan pretplate. Za uslove i detalje se raspitajte na telefon: (011) 651-666 lok. 259 ili 226.

Mogućnosti ima puno, a Vi se odlučite za onu koja Vam najviše odgovara. Ali svakako se odlučite za Računare

- zato što objavljuju najnovije tehničke i poslovne novosti iz zemlje i sveta
- zato što objavljuju prikaze najnovijeg softvera i hardvera
- zato što objavljuju praktična rešenja za probleme na koje nailazite u korišćenju i programiranju računara
- zato što predstavljaju najveću berzu u zemlji za kupovinu i prodaju računarske opreme
- zato što u svakom broju objavljuju dodatak na 32 strane
- zato što tekstove pišu poznati i kompetentni autori
- zato što lepo izgledaju
- zato što često nagrađuju svoje čitaoce stvarima koje su im najpotrebnije ...

Pobrinite se na vreme da na ostanete bez svog primerka "Računara"!

LESKOVAC
Jelenj blok 1 016/43-348

MLADENOVAC
Maršala Tita 16 011/8221-000

NIŠ
Obilićev venac 78 018/ 21-500

NIŠ
Vodova 4 018/ 22-698

NOVA VAROŠ
Rade Stanović bb 033/ 82-101

NOVI SAD
Almaška 3 021/ 51-993

OBILIĆ
KOD PRIŠTINE 038/ 61-648

PANČEVO
Braća Jovanović 22 013/ 43-474

PANČEVO
Dimitrija Tomašića 1-a 013/ 41-590

PANČEVO
Žarka Zrenjanina 3 013/46-082

PIROT
Trg AVNOJ-a bb 010/ 23-657

PODGORICA
Bulevar revolucije 40 081/ 14-918

PODGORICA
Bulevar revolucije 40 085/ 42-165

POŽAREVAC
Maša Pijade 38-40 012/222-164

POŽAREVAC
Stari korzo 3 012/222-073

PRIŠTINA
Kosovskih brigada d-1 038/26-760

PROKUPLJE
Ratka Pavlovića 8 027/ 21-000

SMEDEREVO
Kralja Petra 3 026/222-001

SURČIN
Vojvođanska 73 011/8040-545

VALJEVO
Vojvode Mišića 73 010/ 27-567

VELIKA PLANA
Vaja Jovanića 9 035/ 51-982

VRANJE
Žikica Jovanovića bb 017/ 23-409

ZRENJANIN
Kralja Aleksandra 6 035/ 61-118

ŽELEZNIK
Darinka Radović 28 011/571-484

Tehnologija štampača

QMS laserski štampač u boji

Laserski štampači u boji nisu novoat, ali au koštali i do 50.000 dolera. QMS *ColorScript Laser 1000* spušta cenu tehnologije negde između 10.000 do 15.000 dolara.

ColorScript Laser 1000 koristi elektro-fotografsko beleženje i daje kolorni otisak sa 300 tačaka po inču. Ono što ga čini još privlačnijim je dosad neviđen dizajn mašine koje može da služi i kao crno-beli i kao kolor štampač. To se postiže upotrebom suvog tonera i specijalnog OPC remena (OPC – *organic photoconductor*).

Remen sa okreće ravnomernom brzinom, osam puta u minuti. Kada se štampa slike u boji potrebna su četiri rotacije remena – za svaku boju po jedna. s brojni prolasci laserskog zraka smeštaju celu kolor sliku na remen. Bubanj i valjak za transfer skidaju sliku sa remena i prenose je na medij, koji može biti običan papir nalepnica ili folija. Pošto se remen okreće uvek istom brzinom, *ColorScript Laser 1000* može da štampe dve kolor slike u minuti. Pri štampanju crno-belog dokumente potrebna je samo jedna rotacija i samo crni toner, i tada se štampa prosečno 8 slika u minuti.

Ako zavirnemo u unutrašnjost ovog štampača, naći ćemo 25-MHz 80960CF RISC kontroler. Standardna memorija iznosi 12MB RAM, a može se proširiti do 32 MB pomoću SIMM od 4 ili 8 MB. *ColorScript Laser 1000* pruža emulaciju za *PostScript Level 2*, *Hewlett-Packard PCL (Printer Control Language) 5*, i HPGL (*Hewlett Packard Graphics Language*) jezike za opis stranica (PDL – *page description language*). Interni hard-drajv od 40 MB čuva različite programe PDL emulacija i fontove. Dogradnje njegovog softvera ili dodavanje nove PDL emulacije se svodi na jednostavno kopiranje novih datoteka na hard-drajv, koji takođe služi kao virtuelne memorije pri obavljanju obimnijeg štampanja.

SCSI veza omogućava dodavanje eksternih hard-drajvova, čime se proširuje kapacitet štampača, te može primiti više fontove ili virtuelne memorije. Standardni interfejsi uključuju *LocalTalk* serijske i paralelne portove za štampač. Opcioni *Ethernet* interfejs podržava protokole *Apple EtherTalk*, TCP/IP, *DECnet* i *NetWare*. Specijalna logika ESP (*Emulation Sensing Processor*) skenira sve interfejse i detektuje podatke koji pristižu a zatim analizira podatke i odabira jezik (PDL) koji će obaviti posao. QMS ima vlastiti multitasking operativni sistem – *Crown* – koji upravlja funkcijama štampača i istovremeno može da obavlja dve različita printerska zadatka, koji stižu na različite portove i koristi različit jezik za opis stranice.

Za razliku od termalnih štampača sa voskom koji koristi uvek istu količinu sve četiri suve tonere bilo da štampate crno-belu ili kolor sliku. *ColorScript Laser 1000* troši samo one tonere koji su za sliku potrebni. Tako inteligentna upotreba snižava cenu kolor stranice se 50 ne 45 centi, što nije zanemarivo kod štampanja velikih tiraža. Ako se koristi samo crni toner, cena je 3 do 3.5 centa po stranici.

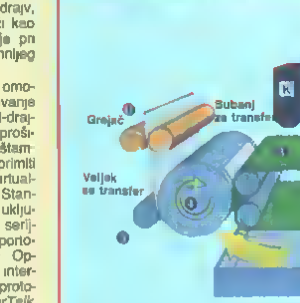
Korisna adresa QMS Inc., Box 81250, Mobile, AL 36689, (205) 633-4300, fax (205) 633-0013

HP (ZA SADA) KAŽE NE

„Hewlett-Packard" ubeđljivo vodi na tržištu laserskih štampača, ali još nema nagovještaja da će krenuti putem QMS-a na tržište laserskih štampača u boji. Njihov predstavnik je izjavio da trenutno proučavaju tehnologiju laserskog štampanja u boji, ali će se u proizvodnju upustiti tek kada to sa stanovišta troškova i kvalitata bude opravdano. HP *ink-jet* kolor štampači se dobro prodaju i za sada se još ne oseća potražnja za laserskim kolor štampačima.

Za ovako oprezan stav kaže predstavnik HP, postoje dva razloga: jedan je pouzdanost a drugi cena. HP smatra da se mora boriti i za jedno i za drugo – da obezbedi kvalitet i pouzdanost svojih štampača, ali i prodaju po prihvatljivoj ceni.

QMS COLORSCRIPT LASER 1000 IZNUTRA



1. Uređaj za naelektrisanje unutar OPC remena ravnomerno prenosi elektricitet na remen.
2. Kontrolni štampača prelazi kroz ravan jedne boje slikovnih podataka. Od tih podataka zavisi intenzitet laserskog zraka koji prelazi preko remena. Dok se to dešava, odgovarajuće mesta na remenu se razelektrišu. Kako se remen kreće gredi se red po red latentne slike sastavljene od razelektrisanih delova delova.
3. Deo remena na kojem je smeštena latentna slika prolazi ispod četiri tonera. Otvare se tonei kaji odgovara upravo nanesenoj boji. Razelektrisani delovi privlače toner, a ostali ga odbijaju. Kad slika u boji, ova operacija se ponavlja dok se na nenesu ave četiri procesane boje. Kompletne slika u boji ne remenu se nalazi nakon četiri rotacije.
4. Bubanj se transfer je naelektrisan i skida sliku sa remena.
5. Razlika u voltaži između bubnja i valjka povlači sliku se bubnja i prenosi je na papir.
6. Grejač pomoću toplote i pritiska fiksira sliku se podlogu.

OBRAZOVANJE

Informatika u školama

Bejzik plaši malu decu

Način na koji se kompjuterska nauka uvodi u osnovne i srednje škole u Americi ne obećava svetlu računarsku budućnost – tvrdi Ted Kaufman predsednik firme „Schemers" koja je razvila novi elementarni programski jezik po imenu *Scheme*. Po njegovim rečima bejzik odvraća decu od programiranja obeshrabrujući njihove prve pokušaje – naime njihovi prvi programčići pisani u bejziku često jednostavno neće da rade.

Zato „Schemers" nudi *Scheme* (čitaj „skim") jezik razvijan na univerzitetu MIT-u koji koncept postavlja između sintakse. Razultat je da daci od samo 12 godina bez problema uče nove pojmove kao što je na primer rekurzija. Kompanija tvrdi da su sedmaci nakon manje od 40 razrednih časove bili u stanju da polože i najteži deo ispita *Advanced Placement*, upravo zato što su koristili *Scheme* umesto tradicionog *Pascala*.

Jezik se već koristi u Evropi i Americi i polako potiskuje *Pascal* u uvodnim kursevima iz programiranja na univerzitetima kao što su Berkli i MIT. Problem je međutim što se na zvaničnom ispitu još uvek insistira na upotrebi Pascala. Kaufman predlaže da se kao uvod u *Scheme* u osnovnim školama na početnom nivou uči *Logo*.

Obuka programera

Microsoft univerzitet: 1000 dolara na sat

Nedavno je u Hongkongu osnovan „Microsoft University" na kojem će se organizovati intenzivni kursevi za profesionalne programera i ljude koji se bave podrškom sistema. Tu će se softveraši moći detaljno upoznati iz prve ruke sa raznim Microsoft-ovim proizvodima. Nastave će biti organizovane u skladu sa specijalnim programom – *Microsoft Certified Professional Program* – koji predstavlja kvalifikacioni sistem za procenu tehničkog znanje. Polaznici stiču uverenje da su osposobljeni da vode obuku o upotrebi nekog Microsoft ovog programa.

Microsoft se nada da će ovim potezom postaviti standard u profesionalnom obrazovanju i rigorozan međunarodni kriteri na osnovu kojeg će softverska kompanije moći da biraju svoj tehnički kadar.

Prvi kursevi su počeli u julu a teme je podrška za *Windows NT* i razvoj *Win32* aplikacije dok će se tokom leta održati još desetak. Oni ne samo da će polaznicima pomoći da izvuku ono najbolje iz Microsoft-ovog softvera već će mnoge pripremiti i za izlazak na specijalni ispit – *Microsoft Certified Professional Exam*.

Ispit traje sat vremena a plaća se $1000. Kažu da oni koji dobiju Microsoft CP kvalifikaciju stiču garanciju za dobro zaposlanja širom sveta. Ish program se u Sjedinjenim Državeme odvija još od 1991.

Korisna adresa Sasha Skonner (Microsoft) Tel +852-804 4261

ZANIMLJIVOSTI

Piratevanje softvera

Ne brinite, ima i gorih od nas

Izveštaj Saveza izdavača poslovnog softvera (BSA – *Business Software Alliance*) neprofitskog udruženja vodećih proizvođača upozorava da softverska industrija zbog piratstva godišnje gubi oko 12 milijardi dolara. Od toga 5.4 miljarde u Azji u Americi 4.5 a u Evropi 1.9. BSA, nadalje navodi da se 1992 godine oko 30% svog softvera na Tajlandu prodavalo i koristilo neovlašćeno. Japan je drugi na listi sa 32% a onda sledi Španije sa 66%. Nemačka sa 62% (Šta mislite ne kojem mestu bismo se mi našli?)

Neki su posle ovog izveštaja i odahnuli – na primer indijska vlada koja se pribojavala rigoroznih kaznenih mera nekih američkih proizvođača. Pokazalo se međutim da su indijski pirati prilično bezazleni u odnosu na konkurenciju jer se na ovu listu uopšte nisu plasirali.

VAŠA NOVA SUPER-TURBO-EXTRA KONFIGURACIJA

Postoji mnogo kompjuterskih dilera, i svi su najbolji, najkvalitetniji i najpouzdaniji. Ako vam se dogodi da vaša nova konfiguracija izgleda ovako (ili malo ozbiljnije), biće vam slaba uteha što se na greškama uči.

Ne učite na najskuplji način.

Poslužite sa nama!

ADACOM COMPUTERS

VAŠ POUZDAN PARTNER

PRODAJA I MARKETING PRO MUSICA ČIKA LJUBINA 12 11000 BEOGRAD JUGOSLAVIJA

TEL: (38 11) 62 92 33, FAX: (38 11) 62 92 33; 62 96 72

TOTAL DESIGN

NA POLA PUTA

Ako se vaše potrebe i mogućnosti još uvek kreću oko 386 klase, matične ploče se Cyrix procesorima Cx486SLC (32/16 bita) i Cx486DLC (32 bita) vam mogu omogućiti da po (gotovo) istoj ceni osetite i nešto od čarolija iz 486 sveta.

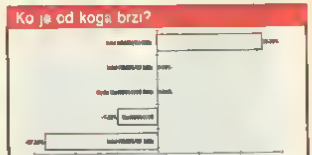
Procesor Cx486DLC se nalazi negde na pola puta između Intelovih procesora i386DX i i486DX. I dok jedni smatraju da se ta sredokraća, zapravo, zove i486SX drugi na ovaj Cyrix procesor gledaju kao na „pojačani 386", dakle kao na i386DX procesor se znatno unapređenim performansama. U Cx486DLC je ugrađen kompletan set instrukcija procesora i486, ali mu za punokrvnu 486 klasu nedostaju dve bitne stvari – matematički koprocesor u celosti i interni keš odgovarajuće veličine. Pošto matematički koprocesor nema ni i486SX, prirodno je što se Cx486DLC ne samo poredi upravo se ovim procesorom, nego se u aplikacijama i odaziva na njegovo ime. Možda bi najtačniji opis ovog procesora glasio „nešto oslabljeni i486SX" – oslabljeni za razliku u veličini internog keša, ali se blagim prednostima na drugoj strani ističe karakteristike ima i procesor Cx486SLC, osim što se spoljnim svetom komunicira preko šesnaestobitne magistrale. Oba modela Cyrix proizvodi u saradnji se firmom Texas Instruments.

Intelova 486 porodica ima interni keš od 8 KB, a procesor Cx486DLC od 1 KB. Osim po veličini, keš memorije kod ovih procesora se razlikuju i po primenjenim algoritmima. Intel u svojim procesorima koristi keš tzv. *write through* tipa (keširaju se samo *read* ali na *write* operacije), a Cyrix *last recently used* tipa, sa faktorom pogodaka od 85 odsto. Nedostatak matematičkog koprocesora Cyrix je pokušao da ublaži ugradnjom aritmetičke jedinice za množenje (nezavisna (celih brojeva) u hardveru. U Intelovim procesorima ovu vrstu množenja obavlja mikrokod, pa su direktni Cx486DLC takmaci, i386DX i i486SX, u operacijama se „golim" brojevima na istom radnom taktu sporiji i do četiri puta. Suštinska razlika je, međutim, u mehanizmu i efikasnosti izvršavanja naredbi. Procesor Cx486DLC u jednom ciklusu dekodira četiri, a i386DX samo jedan bajt instrukcija dok u fazi izvršavanja Cyrix troši jedan a Intel dva takta po instrukciji.

Ovde je, naravno, najzanimljivija pitanje kako se ove sličnosti i razlike odražavaju na performanse procesora, i koliko se one osećaju u radu računara. Cx486DLC/33 MHz je za nekih 27% brži od i386 na 33 MHz, i za nekih 10% od AMD Am386DX na 40 MHz, čime dokazuje svoju izrazitu nadmoćnost u 386DX svetu. U 486/33 MHz klasi, kojoj bi želeo da pripada, Cyrix se nalazi na časnom začelju – prema rezultatima *PC Bench-Mark* testova, za dlaku je sporiji od procesora i486SX na 25 MHz (manji takt!) i prilično zaostaje za i486DX na 33 MHz, koje je brži za 25%. Sve, naravno, ima svoju cenu pa i procesna snaga Cyrix u 486 svetu ima najmanju snagu i – najnižu cenu.

Dogradnja se 386 ne 486 pomoću Cx486DLC procesora, na žalost, nije nimalo jednostavna. On ima drugačiji raspored nožica pa se ne može naprosto ubosti u 386 podnožje, a zahtava i neka posebnosti u BIOS-u. Jedina mogućnost, prema tome ostaje zamena matične ploče. Mi smo nedavno bili u prilici da probamo jednu ovakvu ploču se oznakom AL386-33/40C(ALI). Ploča ima 128 KB (moguće i 64 i 256 KB) sekundarnog *write-back* keša, prima do 32 MB RAM-a (testirano se 4 MB). Ima podnožje za 80387 ili Weitek koprocesor, šest šesnaestobitnih i jedan osmobitni slot – ukratko, tipičan proizvod za 386 klasu.

Brzina (u procesorskim harmonicima) prema rezultatima testa *PC BenchMark*



Intel i486DX/33MHz je brži od Cx486DLC, a Cx486DLC od i386DX/33 MHz i Am386DX/40MHz

Ploču smo ugradili u računar koji (osim i) više časova dnevno koristimo u svakodnevnom poslu, da bismo, poredi ispitivanja „sirove" brzine, došli i do zaključaka o stvarnoj kompatibilnosti procesora i pouzdanosti celog sistema. U gotovo dvomesečnoj intenzivnoj upotrebi ploče sa veoma raznorodnim i zahtevnim softverom nismo uočili ništa što joj ne bi išlo u prilog. Ploča ja pokazala visoku pouzdanost i zavidnu brzinu u odnosu na jednu DTK 386DX/33 MHz ploču čije je mesto zauzela u računaru. Njena cena je 350 DEM a na domaće tržište je donosi firma MEGA. Svojom cenom od nekih 150 dolara, procesor Cx486DLC u potpunosti pripada i386 klasi, dok je po performansama veoma blizak višestruko skupljoj i486 familiji. U kompletu sa matematičkim koprocesorom Cyrix 80387, koji ga još više izjednačava u snazu se i486 procesor Cx486DLC na košta više od 220 dolara. Naravno, ni tada to neće biti baš „ono pravo", ali je u svakom slučaju barem dva puta – jeftinije.

Jovan Regasek

Korisna adresa

MEGA
11070 Novi Beograd, Bulevar Lenjina 6/12
Tel (011) 626-605 629-566

Slicnosti i razlike

	i386	Cx486DLC
Set instrukcija	i386	i486 ~
Dekodiranje instrukcije	bajt po ciklusu	četiri bajta po ciklusu
Izvršavanje instrukcije	dva ciklusa	jedan ciklus
Interni keš	nema	1 K
Koprocesor	nema	nema
Celobrojno množenje	mikrokod	u hardveru

ZANIMLJIVOSTI

Sporovi

Intel i Cyrix pre pomirenja

Još od kraja 1990 godine traje sudski spor između giganta među proizvođačima čipova, Intela, i malenog ali inovativnog Cyrixa. Spor se vodio bilo pravo Cyrixa da prodaje matematički koprocesor tajvanske proizvodnje koji je kompatibilan sa Intelovim čipom ali je ne samo jevtiniji već i brži. Izgleda da bi svađi uskoro mogao doći kraj. Pominanje bi trebalo da se zaključi novim sporazumom između dve kompanije što će se desiti ukoliko sud odluči da odbaci Cyrixovu tužbu kojome se Intel tereti za monopol.

Prvobitni spor je vođen pred takozvanim sudom pošteni januara 1992. a pre nekih više od godinu dana donesena je presuda u korist Cyrixa. Radilo se o unakrsnim licencama između Intela i Cyrixovog snabdevača, firme SGS-Thompson.

Nova rasprava, u nevom speru je zakazana za oktobar ove godine i ticala bi se Cyrixovu tvrdnje da Intel zloupotrebljava svoj monopol te da izvesni Intelovi patenti više nisu važeći i ne mogu se primeniti.

Prema odredbama nedavno potpisanog sporazuma, ako se Intelova tužba iz 1992. ne prihvati, Intel treba da plati Cyrixu 500 000 USD. Ako pak presuda bude donesena u korist Intela, Cyrix će platiti za Intelovu licencu 2 miliona dolara. U svakom slučaju Cyrixu će biti priznato pravo da proizvodi matematičke koprocesore pod dle sada, spor nem licencom.

Inače matematički koprocesori koji se koriste sa 80486SX ili čak kojim 80386 80386SX, 80386, ili 8086-88 kompatibilnim mikroprocesorom obavljaju izvesne složene matematičke proračune mnogo brže zahvaljujući specijalnim algoritmima namenjenim upravo tim operacijama. Procesori klase 486 (izuzetak je 80486SX) i Pentium imaju ugrađen matematički koprocesor.

U praksi matematički koprocesori se ne koriste baš previše – potrebni su samo programima se mnogo grafičkih operacija. Međtim program koji koristi funkcije matematičkog koprocesora radi i deset puta brže nego kada bi se celanjae na mikroprocaser bez koprocesorskih funkcije

TEHNOLOGIJA

Čipovi

Kako više volite: su jednom ili dve kocke čipova?

IBM je počeo ograničenu proizvodnju kockica-čipova, u koje staje više integrisanih kola na isti prostor jer se oni nađaju jedan iznad drugog u slojeve. Otvorena je probna proizvodna linija nazvana *Cubing Process Development Center* „Kockica" se zasniva na tehnologiji slaganja čipova razvijenoj u kompaniji „Irvine Sensors" IBM i „Irvine" su prošle godine potpisali sporazum o zajedničkom komercijalnom ostvarenju ove ideje. Prvi uzorci kockice čipova se očekuju početkom 1994. Koristiće ih i sam IBM – još se ne zna u kojim od svojih proizvoda, ali njihov predstavnik kaže da pristaju svuda gde se priključuju moduli i kartice – dakle kockica bi mogla naći mesta u bilo kojem sistemu.

Korisna adresa Irvine Sensors 714-549-8211 IBM 802-769-2463

Super računari

Da li Japanci to mogu bolje?

„Hitachi" konkuriše IBM-ovoj tehnologiji paralelnih računara, se svojim paralelnim mašinama zasnovanim na RISC čip-setu za koje tvrdi da će biti potpuno kompatibilne sa brže od IBM-ovih.

Računar koriati sve popularniju Hewlett-Packard tehnologiju PA-RISC ali će zbog dupliciranja RISC tehnologije ne jednom jedinom čip-setu snaga mašine biti mnogo puta veća od H-P PA-RISC radne stanice – biće u rangu superkompjutera.

Zahvaljujući otvorenom pristupu arhitekturi sve mašine će prihvatati VOS-3 (Hitachi-jev vlastiti operativni sistem) kao i IBM-ov softver za paralelnu tehnologiju što znači da će upravo primeni nove tehnologije prihvatiti netalike hiljade postojećih aplikacija.

Datum izlaska i cena još nisu poznati mada se govori da će biti jeftiniji od IBM-ovih paralelnih računara. All IBM se neće tek tako predati – u pripremi je računar za opšte namene se 48 čipova, kažu, još brži od Hitachi-jevog

Software NOVAC 3.0

OPCIJE PROGRAMA

- Planirane promene
- Realizovane promene
- Keš promene
- Virmani
- Izvod SDK
- Prenesi stavku
- Adresar
- Kursna lista
- Namena troškova i prihoda
- Svrhe doznake (virmani)
- Backup podataka

IZVEŠTAJ

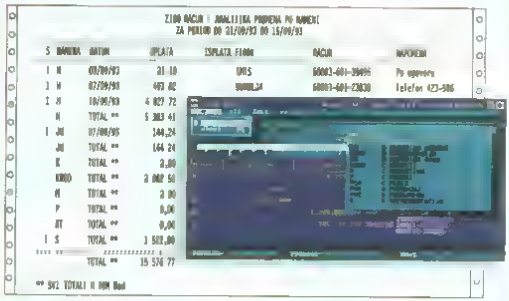
- Dospeva za naplatu
- Očekivani prihodi
- Analitička promene
- Stanje blagajne
- Cash-flow analiza
- Štampanje adresara
- Štampanje kursne liste

ALATI I DODACI

- Predračun kamate
- Aritmetički kalkulator
- Algebarski kalkulator
- Računanje kontrolne cifre
- Datonski kalkulator
- Kontrola pristupa opcijama programa
- Makro tasteri
- Izbor štampača
- Clipboard

Potpuna integrisanost sa ostalim paketima u JDDBE tehnologiji

Kolika je realizacija sektora veleprodaje u prvih petnaest dana? A maloprodaje od početka godine? Koliko nas je koštao marketing u prethodnom mesecu? Kakvi su naši dosadašnji finansijski odnosi sa preduzećem Jugodata? Kada smo platili telefon i koliko? A koliko je to realno u markama a n[illegible]arskom sedmu? Odgovori na ova pitanja su na dohvat ruke - pogledajte vaš zahtev u jednu tabelu i to je sve - ana[illegible]tički izveštaji su pred vama!



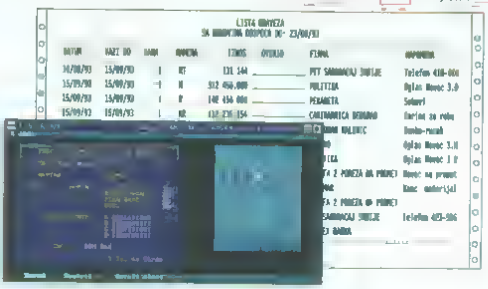
NOVAC 3.0

SVI PUTEVI VODE KA NOVCU-NAŠ JE NAJKRAĆI

Specijalni uvodni popust 10%

199 DM

važi do 15.10.1993.



Unesite unapred vaše obaveze i očekivane prihode. Planirajte i odobravajte plaćanja na osnove izveštaja koje program pripema. Odštampajte virmane i po povratku iz SDK uravnite planirano sa izvođenim. Unesite i sve prihode i rashode u gotovom novcu. I to je sve—baza podataka o vašem novcu je spremna!

Pogled u budućnost!

NOVAC 3.0

Neophodan alat svakom finansijskom direktoru. Omogućava kontrolu nad [illegible] vrsta novca čime [illegible] maksimalizujete profit. Analiza izvršenih transakcija, [illegible] odliva. Upravljanjem vaših [illegible] u gotovini se značajno pove[illegible] u istu valutu obezbeđujete pravo [illegible] vaših poslovanja. Nudimo vam [illegible] koji će vam pomoći da vodite [illegible] u finansijskom [illegible] preduzeća.

jugodata

B. Revolucije 326, 11050 Beograd; tel/fax: (011) 418-326, 419-768, (081) 11-618

TEMPERAMENTNI ŠTEDIŠA

Štampač HP LaserJet 4L je poslednji izdanak već legendarne Hewlett Packard tehnologije. U njega su ugrađena najbolja rešenja ranijih generacija laserskih štampača, ali i po prvi put predstavljene neke nove tehnike - *stand by* režim (Packard ga zove *Intelligent Off*), mnogo ekonomičnije korišćenje memorije i poseban režim štampanja, nazvan *EconoMode*, u koma se štedi toner za 50% – koja će imati odlučujući uticaj na njegov tržišni uspeh.

Jovan Regusek

Paralelno sa modelima vrhunska klase za profesionalnu primenu (LaserJet II+ i III), Hewlett Packard poslednjih godina proizvodi i pojednostavljene modele laserskih štampača za ličnu upotrebu (IIP i IIIP), koji u nazivu modela imaju slovo P (od *personal*). Nedugo nakon promocije najnovije generacije, sa oznakom 4M i 4M (vidi prikaz u „Računarima" 91) Packard je izbacio na tržište i pojednostavljenu verziju ovog modela, ovoga puta lošiko olakšanu da ja u imenu dobila slovo L (od *light* = lagan) umesto P. Sa svojih 7 kilograma težine i dimenzijama 353x362x164mm on je 22% manji i 30% lakši od svog neposrednog prethodnika, modela IIIP, i verovatno najmanji model laserskog štampača koji se može naći na vašem stolu. Prilikom oblikovanja kućišta vođeno je računa o svakom detalju koji utiče na racionalno korišćenje prostora – utikačke gneza za struju i Centronics su toliko uvučena u štampač i tako elegantno rešena da na zauzimaju nimalo mesta na stolu.

Skroman u zahtevima, izdašan u onome što pruža: HP LaserJet 4L nema konkurencije u lakoći korišćenja

INSTANT TEHNOLOGIJA

Štampači HP 4 i HP 4L su programski kompatibilni i imaju manje-više iste mogućnosti, ali se u tehničkim detaljima i rešenjima toliko razlikuju da bi se teško reklo da pripadaju istoj porodici. U 4L je, pre svega, ugrađena nova, ali jednostavnija Canon mašina rezolucije 300×300 tačaka (HP 4 ima 600 tačaka), znatno slabiji i sporiji procesor Motorola 68000 (u HP 4 RISC procesor i960) a izostavljene su ave pogodnosti za povezivanja i proširivanje štampača. Zajednički su im samo BiTronics paraleni interfejs, koji je potpuno kompatibilan sa Centronics interfejsom ali omogućuje protok podataka u oba pravca i ima desal puta veću propusnu moć i RET (*Resolution Enhancment*) tehnologija koja modifikovanjem veličine laserske tačke popravlja kvalitet ispisa slova. I pored oznake 4 na kućištu pred sobom ipak nećeta imati famoznu četvorku, jer većinu njenih ključnih osobina i pogodnosti oiakšani model nema.

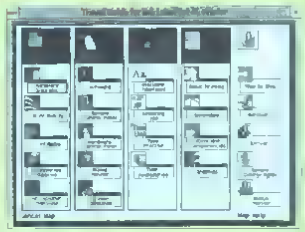
Sve ovo naravno, ne znači da model 4L nije vredan svaka moguća pažnje. U pitanju je poslednji izdanak već legendarne Hewlett Packard tehnologije, u koji su ugrađena najbolja rešenja ranijih generacija laserskih štampača, ali i po prvi put predstavljene neke (barem za ovu oblast) nove tehnike - *stand by* režim (Packard ga zove *Intelligent Off*) mnogo ekonomičnije korišćenja memorije i poseban režim štampanja nazvan *EconoMode*, u kome sa štedi toner za 50% – koje će imati odlučujući uticaj na njegov tržišni uspeh.

U štampač je, pre svega, ugrađan novi. Instant sistem grejanja, koji radnu temperaturu postiže veoma brzo: reklo bi se gotovo trenutno. Ova tehnička inovacije je utrla put jednom potpuno novom konceplu (koji je inača dobro proveran u tehnologiji proizvodnje televizora, video-rekordera i Hi-Fi opreme) komunikacija sa lasarskim štampačem. Štampač je stalno uključen u struju (nema čak ni mrežni prekidači) i na posao čeka u stanju mirovanja (troši samo 5W energije, potpuno je bešuman, ne zagreva okolinu) da bi se automatski probudio čim mu stigne prvi podatak na port. Aktiviranje štampača je praktično trenutno i potpuno neprimetno – korisnik ima utisak da je on stalno spreman za rad. Ova pogodnost obezbeđuje posebnu lakoću korišćenja štampača i ima čitav niz praktičnih prednosti – fontovi koje jednom spustite u štampač (*download*) ostaju u njemu dok ne nestane struje. Štampanja ne zahteva nikakve pripreme štampača – ako u magacinu imate papira, dovoljno je da kažete PRINT. Nakon petnaest minuta neaktivnosti, štampač se automatski povlači u stanje mirovanja. Ovako postavljen, štampač prestaje da bude periferni uređaj i postaja integralni deo personalnog računara.

POD SOFTVERSKOM KONTROLOM

Mala četvorka, za razliku od velike, ima krajnje pojednostavljen kontrolni panel. Umesto višefunkcijske tastature i alfanumeričkog displeja u njega je ugrađen jedan jedini taster i samo četiri svetleće diode za indikaciju najelementarnijih funkcija i grešaka: ERROR (otvoran štampač, problemi sa tonerom), PAPER (nestalo papira ili papir zaglavljen), DATA (podaci još u štampaču ili čekanje da papir bude ubačen ručno) i READY (štampač spreman za rad ili prijem podataka u toku).

O mogućnostima ovog štampača međutim, ne treba suditi po njegovom krajnje oskudnom kontrolnom panelu. Ovo je prvi Hewlett Packard model čijim radom – zahvaljući BiTronics paralelnom interfejsu – u potpunosti upravlja računar. O kontroli parametara i rada štampača brine mali rezidenti program po imenu *Status Monitor*. Njegova osnovna namena je da bdi nad komunikacijom računer-štampač i da upozori korsnika čim nešto nije u redu, bez obzira da li je štampač u aktivnom stanju, dakle u fazi prijema podataka i štampanja ili u stanju mirovanja. On reaguje na sve uobičajena problema u toku rada se štampa-



Pod punom softverskom kontrolom: Naslovne strana i osnovni meni programa *Travel Guide*

čem, o kojima na drugim modelima brne kontrolni panel, ali i na njegov opšti status – ako se on isključi iz struje ili otkači sa Centronics kabla.

Pored nadgledanja reda, *Status Monitor* služi i kao *hot key* (Ctrl+Alt+E) šel – kontrolni centar za poziv odgovarajućih softverskih alatki za podešavanje štampača. Iz njega se mogu aktivirati dva osnovne alatke. *HP Explorer Travel Guide* i *HP Explorer Remote Control Panel*. *Travel Guide* je vodič kroz mogućnosti i pogodnosti štampača, kao i help sistem za njegovo podešavanje. *Remote Control Panel* služi za daljinsko upravljanja svim podesivim opcijama štampača – kontrola lonera, gustine i tehnika štampe, izbor formata strana i broja redova na njoj, izbor osnovnog pisma i kodnog rasporeda – i koristi se sa onim aplikacijama koje nemaju odgovarajuće drajvere za rad sa ovim štampačem. Osnovne ideja je da se kontrola štampača potpuno prepusti odgovarajućim drajverima, i ona je u u drajveru za *Windows 3.1* dosle dobro ali ne i do kraja iskorišćene – izostala je, sa našeg stanovišta, jedna ključna

opcija, kojom čemo se više pozabaviti u odeljku o fontovima.

TONER: ŠTEDLJIVA ŠTAMPA

HP 4L koristi posebnan tip kasete za toner (cena = 85 dolara) all sa istom vrstom sitnozrnastog praha kao i HP 4, koji je posebno zahvalan za RET tehnologiju popravljanja kvaliteta ispisa. Proizvođač dakianše radni vek na uobičajenih 3000 strana uz pokrivenost od 5%, i po prvi put uvodi posebnan režim štampe za sledeću skupocenog crnog praha. *EconoMode* štedi najmanje 50% tonera po strani a namenjen je za štampanje probnih kopija i uopšte dokumenata kod kojih kvalitet otiska nije presudan. U štedljivom režimu rada – u koji se prelazi dovoljno lako da jednu stranicu za proštampa možete da štampala u njemu a odmah potom i normalno – otisak je pnlično bledi ali do voljno čitljv. Slova izgledaju kao da je u njima štampana lak svaka druga tačka (grafičan bi rekli „kroz rastar od 50%"). Režim štampe na utiče na brzinu rada štampača već samo na izgled otiska. Kaseta je pravljena od materijala koji se mogu reciklirati a Hewlett Packard usrdno moli kupce ovog modela da mu vraćaju prazna kasete o njegovom trošku. To je navalo konsnike da posumnjaju da se radi o toneru za jednokratnu upotrebu i da ga nije moguće puniti novim prahom kao ranije modele. Ako se isposlavi da je ovo tačno domaći vlasnici ovog štampača će biti u pnlično nazavidnom položaju – novi toneri su na samo skupji nego se do njih i mnogo teža dolazi.

Univerzalni magacin za papir prima sto listova u četiri formata – *executive*, *letter A4* i *legal-sized* (Dosta veliki napredak u odnosu na ne tako davno vreme kada smo se dovijali da *letter* kasete na amerikom modelu Series II+ konstimo za A4 papir.) Magacin se prilagođava podasavanjem odgovarajućih graničnika, a proizvođač preporučuje da se svaka promena formata saopšti u drajveru za štampač da na bi došlo da zaglavljivanja papira. Štampač prima i kovarte i folije pri čemu se preporučuje da se ubacuju ručno.

Mehanizam dostavljanja papira i magacin za njegovo skladištenja su dosta zahvalno pojle za ušteda svaka vrsta i Packard se nije ustručavao da uštedi koliko je to moguće. Kaseta ničim nije zaštićena se donje strane i golovo dodiruje površinu na kojoj se nalazi štampač a njen kapacitet od 100 listova se mnogim konsnicima može učiniti sumnjivo malim. Naravno samo korisnik može znati kojim intenzitetom koristi svoj štampač. Mi imamo utisak da je za onu kalgoriju kojoj je štampač namenjen – pojedinačna upotreba kod kuće ili u kancelariji – ovaj kapacitet sasvim pristojan i da ga njegovi vlasnici neće osećati kao naročitu smetnju.

MEMORIJA: OVOGA PUTA DOVOLJNO

Prve kupci novih modela HP laserskih štampača su obično nakon prvih ushićenja novom tehnologijom kratali u besomučnu potragu za memorijskim proširenjem. Pokazalo se do sada više puta da je u bazna modela ugrađivano premalo memorija za štampanje složenih grafičkih strana A4 formata. Kupci modela 4L su po svemu sudeći pošteđeni ove nepnjatnosti. U mašinu je doduše ugraden samo 1 MB memorije, koji štampač koristi za svoje režijske potrebe i za formiranje strane, ali je ta količina uz posabna softverska tehnika obrade slike i fontova, dovoljna za sve vrste šlandardnih poslova. Ova tehnika koju HP naziva *Memory Enhancement Tehnology* (MEI) praktično udvostručuje raspoloživi memorijski prostor različitim malodarna kompresije fontova i podalaka. Packard naravno dopušta situacija u kojima ni to ne mora da bude dovoljno, i šalda se proširenja memorija za još 1 MB ne može izbeći. Mi prilikom štampanja iz programa *CorelDraw 4* i *PhotoShop 2.5* nismo osetili potrebu za dodatnom memorijom.

U radu sa šekstom MEI komprimuje kako fontove koji se upisuju u štampač tnajno (*download*), tako i one koji mu pristižu u toku seme štampe. U radu sa grafikom MEI analizira čitavu stranu i u zavisnosti od nalaza primenjuje odgovarajuću kompresionu tehniku za štampanje A4 strane punog formata u rezolucije 300 tačaka po inču. Softver, pri tom koristi dva originalna metoda za kompresiju podataka – *Image Adapt* (prilagođavanje slika) i *Page Protect* (zaštita stranice). *Image Adapt* komprimuje sliku uz male gubitka u grafičkim detaljima, i zato je korisnicima ostavljena mogućnosl da biraju strategiju koja im najviše odgovara. *Image Adapt* može da radi automatski (*auto* – samo kada strana ne može da stane u memoriju) stalno (*on* – brže štampanje grafika uz gubitka u kvalitetu i tamo gde to nije neophodno) ili nikako (*off* – štampe se samo deo slika koji staje u mamoriju).

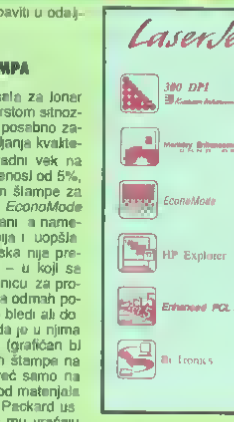
HP 4L štampa stranicu uporedo sa formiranjem njena slike u memoriji. Prilikom obrade složenijih strana može se dogoditi da elektronika posustane za mahanikom ne uspevajući da joj doluri sva podatke na vreme i strana izlazi oštećena uz upozorenje na grešku. Za ovu vrstu problema zadužen je *Page Protect* mehanizam. Kada je on aktiviran štampanje počinje tek nakon formiranja strane u memoriji štampača. Pošto se lime gubi dragoceno vreme predvidena je i mogućnost automatskog rada kada štampač sam procenjuje da li je neku stranu potrebno na ovaj nacin zaštiti ili ne. Oba softverska mehanizma i *Image Adapt* i *Page Protect*, su podešena na automatski rad a režimi se po potrebi mogu menjati ili preko **Printer Setup** menija iz bilo koja aplikacije ili pomoću programa *Remote Control Panel* iz *HP Explorer*-a.

ZAKRPA NA DRAJVERU

Windows drajver za štampac nema mogucnosl izbora kodne strane za interne fontove. Drajver je, programski podešen za kodnu stranu *Windows 3.1 Latin 1* sa oznakom *19U* čima je praktično uskraćen pristup internom trezoru fontova svim konsnicima a medu njima smo i mi koji na mogu da pronadu svoja slova u ovom kodnom rasporedu.

Kodna strana je zapisana u dalotecl HPPCL5E3.DLL. Dovoljno je (korišćenjem *PC Tools* ili *NU* programa) sining *19U* za manti slringom *9E* (oznaka za kodnu stranu *Windows 3.1 Latin 2* koja se konsli u varzli za srednju i isločnu Evropu i koja nam savršeno odgovara) i pul do naših slova u internim fontlovima će biti otvoren. Ako se prepravka drajvera obavi pre instalacije zamenu je dovoljno izvršiti samo na jednom maslu dok je posle instalacije lo neophodno učiniti za svaki font posebno. U lslo vreme bi valjalo zamenili i string *(W1)* slringom *(WE)* koji se u lsti fontova koristi kao indikalor izabrane kodna strana.

Ovo rešanje na žalosl predstavlja grubu prevaru i vredi samo do prvog poravnavanja desna margine. *Windows* ča i dalje smatrali da radi sa kodnom stranom *19U* i iz nje uzimati šinna slova, pa desna ivica kod proporcionalnih pisama neće biti uravnala kako valja. Možda bi se ozbiljnijom analizom drajvera našao lak i za ovaj problem. Ako vas ideje koja smo iznell u ovom lekstu dovedu do rešenja bice nam drago ako budala osetili potrebu da ih podelita sa nama.



Sva osobine na jednom mestu: Test strana sa tehnološkom ličnom kartom

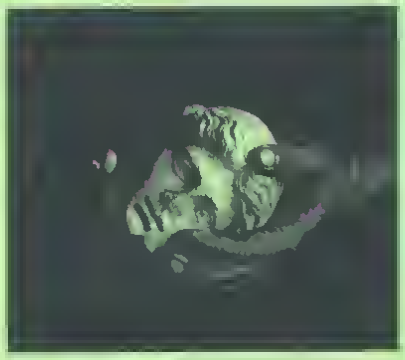
FONTOVI: SUPA ZA ŽDRALA

U programskim mogucnostima HP 4L nije naročito pnkraćen u odnosu na ostala članove porodice 4. U njega su takoda ugrađena tri programska jazika – Enhanced PCL 5 (komandni jezik za upravljanje štampačem) HP-GL/2 Graphics Languge (*de facto* industrijski standard za vektorsku grafiku) i PJL (*Printer Job Languga* – jezik za opis posleva).

Izbor internih fontlova je nešto oskudniji (26 slovnih likova iz familija Couner, Letter Gothic CG Timas, Coronet, Univers, Univers Condensed, Antiqua Oliva, Albertus i Windings) ali pažljivo odabran da pokriju široku opseg potreba u štampanju. Iaksla. Uz obavezna serifna besarifna i neproporcionalna u štampaču se našlo mesta i za naslovna pa čak i za jedno ukrasno pismo. U pilanju su Agfa Inlellifont konturni fontlovi čija se veličina može kretati, u koracima od po 0.25 izmedu 0.25 i 999.75 punkta (graličkih lačaka) kod proporcionalnih odnosno 0.10 i 576 pica kod neproporcionalih pisama.

Pnstup internim fontlovima iz *Windows* okruženja ne zahteva gotovo nikakav trud korisnika. Nakon instaliranja odgovarajuceg drajvera u listi fontova će se sa praporaštljivom ikonicom štampača naci i svi interni fontovi potpuno ravnopravni sa fontovima koji su iz ostalih izvora ugrađeni u *Windows*. Batem u Microsoft aplikacijama kao što su *Word Excel Write* i verovatno i svim ostalim koje prilikom lesta nisu bile instalirane na našem računaru. Interna fontove poznaje i *Ventura 4.1 for Windows* ali ih mnogi programi posebno iz oblasti stonog izdavaštva grafičkog oblikovanja i obrade slika (*Corel QuarkXpress PhotoShop*) potpuno ignorišu jer iz razumljivih razloga žele de zadrže potpunu kontrolu nad generisanjem slova. Pošto je štampač HP 4L

namenjen prvoj a na drugoj vrsti aplikacija, navedenu činjenicu ne smatramo nedostatkom, nego je naprosto, samo konstatujemo. Ekranski fontovi ne spadaju u standardnu softversku opremu, ali se mogu besplatno dobiti od proizvođača (*TrueType Screen Font Pack 4L*). *Windows* se srećom snalazi i bez njih, rekonstruišući dosta vamo njihov izgled.



Štampanje grafike. Izuzetno pokrivanje crnih površina

Ugrađeni fontovi se javljaju u preko dvadeset kodnih rasporeda, među kojima se nalaze i dve od interesa za naše govorno područje – kodne stranice 852CP i 1250 obuhvataju i sva slova srpske latinice. U verziji za centralnu i istočnu Evropu, *Windows* prilikom prikaza teksta koristi za sve interne fontove kodnu tabelu 1250, i na ekranu se javljaju naša slova savršeno usklađena sa tastaturom.

Ovaj koncept, na žalost, nije razvijen do kraja. Štampaču se može poslati **samo** kodna strana 19U (ona koju u Microsoft-u zovu *Windows ANSI*) i – nijedna druga. Drajveri koji dolaze uz štampač uključujući i najnoviju verziju od 30.04.93. **ne omogućuju** ručni izbor kodne strane, niti su dovoljno inteligentni da na osnovu konfiguracionih datoteka, iz promenljivih **country** i **language**, sami zaključe koja je kodna strana aktivna u računaru. Kao u poznatoj basni o gozbi lisice i ždrala, na kojoj je bilo obilja hrane a ždral ostao gladan, internim fontovi su tu, ne dohvatu ruka sa svim što je neophodno da bi se koristili u našem i mnogim drugim jezicima u svetu, ali im se praktično ne može pristupiti jer HP nije razvio mehanizam rada sa kodnim stranama. Drajver se, hakenisanjem i krpljenjem koda, tako može navesti da umesto kodne strane 19U proziva neku drugu pa i 9E (1250 – *Windows 3.1 Latin 2*) koja je od interesa za nas, ali se time kod proporcionalnih pisama ništa puno dobiti, jer se tabela se šinnama slova i dalje odnosi na kodnu stranu 19U.

Neko će reći da je uz *TrueType* fontova, i efikasne mehanizme kojima *Windows* njima upravlja pristup internim fontovima više akademsko nego praktično pitanje, ali bi se autor ovih redova teško sa tim složio. Fontovi se ograđuju u štampač da bi omogućili brži, elegantniji i kvalitetniji rad i velika je šteta što im HP nije obezbedio bezbolniji pristup. Potrebni podaci su se uz malo dobre volje mogli ugraditi u postojeći drajver, ali bi sa isto tako mogli i (zašto da ne) direktno uzimati iz štampača preko dvosmernog paralelnog interfejsa. To bi bilo pravo potpuno rešenje.

Pristup internim fontovima iz DOS-a nije ništa bolji, ako ne i gori od pristupa iz *Windows*-a. Priključeni su drajveri samo za nekoliko aplikacija (*Lotus 1-2-3 for DOS R2.3/2.4 Wysiwyg Driver, Word Perfect*). Pored teškoća sa kodnim stranama u DOS aplikacijama se javlja još jedan problem – usklađenost printerskih i ekranskih fontova. Preko *Remote Control Panel*-a se doduše mogu izabrati i osnovni font i njegova kodna strana, ali samo za jedno pismo. Nema puno nade ni u pisanje sopstvenih drajvera (tako je poziv kodne stranice krajnje trivijalna stvar) jer i ovde nedostaju podaci o širinama slova. Sva ovo naravno ne znači da konsnici ovog štampača nemaju mogućnost štampanja lakstova na svom matemjem jeziku već samo da ne mogu da koriste ono što već postoji u štampaču i da se u tome moraju snalaziti kao i dosad. Ovo se odnosi na sve modele iz serije 4, a ne samo na 4L.

UŠTEDE BEZ POSLEDICA

Preporučena cena od 845 dolara (u proteklim vremenima verovatno i niže – kod nas nešto ispod 2000 DEM) je gotovo šokantno niska za proizvod s namenom i klase kao što je HP 4L. Kako je Peckard uspeo da postigne tako vitku cenovnu liniju? Sigurno ne bez ozbiljnog stezanja tehnološkog kaiša. Oni rezultati kure mršavljenja koji se mogu videti već na prvi pogled – u štampač je ugrađeno samo 7 kg materijala a vodano je računa da se izostave čak i teko trivijalni detalji kao što je mrežni prekidač ako nisu neophodni – nemaju uticaja na funkcionalnost i kvalitet mašine. Pretpostavljamo da se najviše štedelo na mehanici štampača, ali se tu poslednje ne mogu videti tako brzo, nego tek nakon više meseci pa i godina upotrebe. Ako je međutim verovati onome što proizvođač iznosi u zvaničnoj dokumentaciji čak 90 odsto delova ove mašine može da ide na ponovnu ugradnju nakon što štampaču prođe radni vek. Ipak jedno je sigurno: primanjene uštede nisu ostavile nikave posledice na kvalitet rada. Sa 4 stranice u minuti HP 4L zasigurno nije brzinski šampion (našto je sporiji od modela LaserJet III), ali za klasu 300 tačaka po inču daje odličan otisak. Nova Canon mašina, finiji toner i RET tehnologija naročito popravljaju sposobnost poknvanja velikih crnih površina. To je do sada bila slabe tačka svih pa i ranijih modela Hewlett Packard laserskih štampača.

HP 4L je štampač izuzetno skromnih zahteva i veoma izdašan u onome što pruža. Na stolu zauzima malo mesta, savršeno se uklapa u računarski sistem, ne traži posebnu pažnju i pripremu za rad i spreman je za štampanje u svakom trenutku. Štedi toner i struju sa skolškog stanovišta je potpuno bezbedan i što je najvažnije ima kvalitet otiska koji je u vrhu za klasu 300×300. Za jedan mali kućni i kancelarijski štampač Hewlett Packard ranomse i cena od 845 dolara – više nego dovoljno.

Korisna adresa
Micro Anime
11000 Beograd, 29. novembra 71
Tel. 011 343 940

SAD ILI NIKAD

Nema boље prilike za kupovinu novog hardvera od trenutke kada se smenjuju tehnološke generacije ili, naprosto, familije proizvode. Za one koji razmišljaju o nebavci laserskog etampača, kucnuo je previ čae.

Sa svojom gotovo neverovatnom cenom od 845 dolara Hewlett Packard je uneo pravu pometnju na tržište laserskih štampača klase 300 tačaka po inču. Proizvođači moraju ne samo da sa potrude da održe tehnološki korak sa novim HP proizvodom nego i da ozbiljno potrese cene svojih postojećih modela da bi privlačnom ponudom konkurenciji ono što im u odnosu na konkurenciju nedostaje u postojećoj tehnologiji.

Sticajem okolnosti ne našem stolu se ovih dane uz neverovatnu malu HP četvorku našao i jedan ta kav štampač: model LP60 koji potpisuje nemačka fi ma AEG Olympia sa dugom tradicijom između ostalog i u proizvodnji pisaćih mašina.

Štampač predstavlja tipičan proizvod pedantne nemačke tehnologije koja vodi računa i o detaljima koji drugima ne padaju ne pamet baš previše često. Proizvođač drželi se maksima da je zdravlje najveće bogatstvo sa ponosom ističe da je u ovaj model ugrađen ozonski filter za zaštitu od zračenja. Ekologije predstavljaja najnoviji trend u proizvodnji kompjuterskih uređaja e ovaj model je već napravljen u skladu sa (strogim nemačkim) standardima za zaštitu zdravlja. Ne možemo naravno biti sigurni da su modeli koji nemaju ovakav filter opasniji po zadravje već samo konstatujemo jednu pogodnost ovog modela do koje je njenom proizvođaču veoma stalo.

Sa dimenzijama 458x386x177 mm štampač AEG Olympia LP60 spada u manje (ali dobro nabijene težine 15 kg) modele na tržištu. Kućište samo po sebi ne zauzima puno mesta na stolu ali štampač ipak traži prilično prostore zbog magacina za papir (kapacitet 250 listova) koji je tipično za ovu vrstu uređaje smešten sa strane. Koristi laserskiu mašinu Canon u pa (u kasetu sa tonerom je smešten i selenski valjak) i radi u standardnoj rezolucij 300×300 tačaka po inču. Ima paralelni i serijski interfejs isporučuje sa sa 1512 K RAM-a (dovoljno za štampanje grafika) pa čamu se memonja može u koracima od po 1 MB proširiti na 4,5 MB. Za zagrevanje na početku rada je potrebno 60 sekundi a deklarisana brzina je 4 stranica u minuti. Grejač nije neročito racionalan u konšćenju struje – deklarisana potrošnje je 550 W.

Štampač LP60 softverski emulire popularni Lajer Jet IIP – prvi model koji je Hewlett Packard proizvse za ličnu upotrebu – i po svojim mogućnostima i performansama spada upravo u tu kategoriju proizvoda. U njega je ugrađeno 14 rezidentnih fontova a mogu se naravno koristi i softverski fontovi iz računara ili kartridže. Uz LP60 sa ne ugrađuju nikakvi drajveri za štampač niti će vam oni nedostajati – ne možemo da zamislimo aplikaciju koja ne podržava PCL, *de facto* industrijski standard za kontrolu štampača, i koja se isporučuje bez odgovarajućeg drajvera.

U pitanju je dakle model iz sasvim standardne (u najboljem smislu te reči) ponude za klasu 300 dpi koji se u obilju sličnih proizvoda izdvaja pedantnom i preciznom izradom. I svojom veoma prviačnom cenom. Uz obezbeđeni servis. Adacom nudi ovaj štampač za samo 1700 DEM a u paketu sa Adacom mašinom 486 klase on će svoje kupca koštati samo 1300 DEM. Laserske tehnologije zaista, nikada nije bila pristupačnija niti trenutak za njenu nabavku povoljniji – cena drastično padeju a performansa ostaju iste. Ako tragate za štampačem za ličnu upotrebu ili za finije kancelarijske i dokumentarističke poslove, kucnuo je pravi čas da se odlučite.

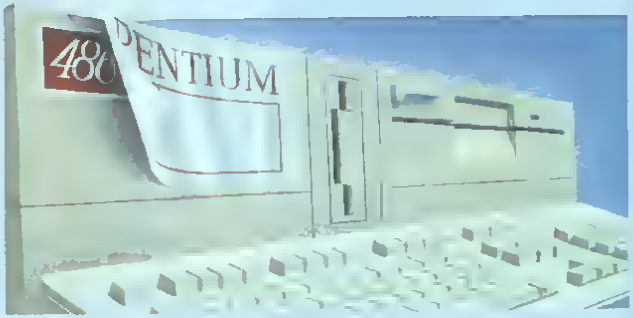
J.R.

Korisna adresa
Adacom
11000 Beograd, Cika Ljubine 12
Tel. (011) 629-233, Fax. (011) 629-672

ČAROLIJA STVARANJA

PENTIUM IZ TEMELJA MENJA PC SVET

Neizmerna sirova snaga Pentium proserora nije sama po sebi dovoljna. Da bi se ona oslobodila, potrebno je iz temelja preoblikovati arhitekturu personalnog računara i stvoriti odgovarajuće operativne sisteme.



Ugradnja superskalarnog 66-MHz procesora u arhitekturu koja je prvobitno projektovana za čip 8086/8088 i 8 bitnu I/O sabirnicu se može uporediti sa Faranjevim motorom u Model-T šasiji motor radi, all se na može reći da baš „šapuće" Pentium će podići cenu svih podsistema. Da bi stvar bila još složenija i softversko okruženje prolazi kroz velike promene pet-sest 32 bitnih operativnih sistema se nadmeće za dominaciju na sistemima sa Intelovim čipom a mijatunizacija host sistema nemece mikrokompjuterima nove i sve teže uloge. Po sudujući koncepcijske rešenja *mainframe* računara mini računara i radnih stanica – i odvajući se na neke sasvim nove pionirske koncepte – proizvođači stvaraju sledeću generaciju PC kompatibilnih računara koji će značiti veliki korak napred u svetu mini-računara.

Ovo se najodlje vidi na primeru servera. Pentium je toliko moćan da po sirovoj brzini prevazilazi neke *mainframe* računare. Ipak ako se nađe u stegama konvencionalnog PC dizajna dobija se sistem koji ne pruža ni performanse ni pouzdanost kakve treba da ima centralni deo jedne velike mreže. Da bi se povećala kritična I/O brzina inženjeri uvode nove sabirnice i podsistema za skladištenja. Osim toga oni koriste tehnike kao što su redundancija korekcija grešaka i dijagnostike da bi 80x86 sistemima obezbedili pouzdanost i mogućnosti upravljanja tipične za host mašina.

Prvi Pentium sistemi nisu u stanju da iskoriste sve prednosti novog mikroprocesora. Kao i kod prelaska sa 386 na 486 stari dizajn se prilagođava novom bržem procesoru. Među tim lokom sledece godine se očekuje veliki broj „pravih" Pentium sistema.

NOVI HIGH-END

Prvi Pentium sistemi ce biti bogato opremljeni, jer se pretpostavlja da prvi najnestrpljiviji kupci neće marati za cenu. Teko ne pnmar. Pentium sistemi mogu imati VRAM (video RAM) grafičke akceleratora, ugrađene SCSI-2 kontrolere CD-ROM drajvova podršku za multimedije preko DSP (*digital signal processor*) ili posvećenih audio/video čipove. U stvari, pojava Pentium sistema najavljuje jedno novo poimanje „standardnog" PC računare jer će se proizvođači utrkivati da ove sisteme opreme sa što više skupih „krpica".

Pitanja je, međutim, da li će ova „deluxe" komponenete zajedno sa Pentium sistemima da se presele i na masovno tržište. Nil Margulis iz kompanije „S3" koja proizvodi video-čipova tvrdi da će nakon masovne ugradnje takve specijalne komponente postati jeftinije. „Većina tih komponenti će sa ubuđeće smatrati standardnim delom Pentium sistema i ljudi neca ni hteti da ih kupuju bez toga." Drugi opet misle da će se u budućnosti ipak prodavati ogoljeni Pentium sistemi zbog osetljivosti kupeca na cenu.

Opšte je mišljenja da će u prvo vrame Pentium naći najveću primenu kao serverska platforma. ali bi sa ovo moglo pokazati samo delimično tačnim. Mark Garvar iz kompanija „Tncord Systems" kaža da će Pentium nako vreme biti suviše moćan procesor, ali oni ipak planiraju da ga ugrada u svoj *ES5000 PowerFrame* server. Garvar ističe da će Pentium serveri povećati propusnu moć u mreži samo ukoliko se adekvatno unapređanja postigne i u I/O performansama.

Sa druga strana, za baze podataka i druge aplikacione servera koji se konste u klijent/server okruženju brži mikroprocesor i memorijski podsistem znače veliki napredak. „Sa Pentiumom će mnogo više dobiti korisnici koji rade sa SQL serverima i programima *Notes* i *Oracle* u OS/2 okruženju nego oni koji rade sa *Net Ware* mrežnim softvarom" ističe Dejvis Filds predstavnik firme „Parallan Computers"

Prvi Pentium sistemi su uglavnom projektovani tako de se mogu dograditi. Kompleks procesor/memorija se može optimizovati za Pentium ali je cell sistem predviđen za 386 ili 486. „Prvu generaciju 486 sistema su činile skrpljene mašine – procesor 486 sirpan u kućište optimizovano za 386" kaže Džim Metos, iz „Hewlett Packard"-a. „Jelo će se desiti i sa Pentiumom trenutno svako živi žun da sklopi sistema zasnovane na 466-tici koji su deleko od optimalnih." Međutim, do kraja 1993 ili početkom 1994 kupci će moći da vida i prve „prave" Pentium sistema – mašine u potpunosti zasnovane na Pentiumu.

PRAVI PENTIUM SISTEMI

Definisati „pravi" Pentium sistem uopšte nije tako. Jedan kritanj bi mogao biti potpun 64-bitni dizajn. all je to nemoguće iz tehničkih i ekonomskih razloga. Na primar 64-bitna varzija ISA/EISA sabirnice ne bi imala nikakvog smisla. Čemu podizati složanost i troškove ako uobičajeni dodatni uređaji, kao što su modemi od toga nemaju niša?

Umesto toga Pentium standard će verovatno kombinovati više sabirnica, brzih interfejsa i viši stepen integracija na matičnoj ploči nego 486 mašina. Desktop sistemi će imati ubrzanu grafiku stereo audio i druge multimedia funkcija. Serveri će pružati visoko-pouzdan rad dijagnostiku i upravljanja sistemom.

Ono što izdvaja Pentium sistema od njihovih prethodnika je memorijski podsistem 64-bitni memorijski interfejs radi na 60 i 66 MHz. Pri takvim brzinama današnji DRAM zahteva ekstemi keš, kako bi procesor stalno dobijao kod i podatke. Većina sistema će konstiti keš od 256 KB ili veći koji će verovatno činiti smhroni SRAM (statični RAM) čipovi što će obezbeđivati prosečan faktor pogodaka veći od 90%.

Sa tako visokim postotkom pogodaka, sta za od kaša do glavne memorije ne mora da bude šira od 32 bita. HP procenjuje de bi 64-bitni DRAM unapredio performanse samo za 5% jer šira sabirnica pomaže samo kada nedostaje keš. S obzirom da se 64-bitna memonja može konfigurisati i proširivati samo u koracima od 8 MB poboljšanja od 5% ne opravdava smanjanja fleksibilnosti RAM-a.

Tom Benot jedan od menadžera kompanija „Micronics" koje proizvodi matične ploče ima drukčija stanovište. On kaže da „sve što ima 32-bitni memorijski interfejs nije pravi Pentium sistem." „Micronics" na svojim Pentium matičnim pločama koristi 64-bitnu memonju jer to na podiže cenu sistema, a čak i 5% bolje performanse mogu mnogo da znače krajnjem korisniku.

Što se tiče nove RAM arhitekture proizvođači još uvek čekaju razvoj događaja. Za sada tradicionalni hijerarhijski dizajn keša može premostiti razliku u propusnoj moći procesora i memorije, a i jeftiniji je. Ali sistemi će uskoro zahtevati brži DRAM da bi nadoknadili nedostatak keša.

HIJERARHIJSKE SABIRNICE

Sledeći element koji definiše prave Pentium sistema jeste upotreba lokalna ili „mezanin" sabirnice. Lokalna sabirnica predstavlja direktan produžetak procesorski linija i radi na istoj frekvenciji kao VESA sabirnica sa procesorom 486. „Mezanin" sabirnica je nešto između lokalna i I/O sabirnice njene su linije odvojene baferom od linija procesora i radni takt joj može biti proporcionalno manji od takta glavnog procesora (½ upola ili za dve trećine spo-

nji). „Prednosti brže sabirnice su očigledne" kažu poznavaoci. Gotovo bez ikakvog povećanja cene sistema, dobijate mnogo bolje performanse.

„Intel" podstiče proizvođače da prihvate njihovu PCI (*Peripheral Component Interconnect*) specifikaciju i čip-set za „mezanin" sabirnicu u Pentium sistemima. PCI sada predviđa 32 bita i radi dvepul sponje od eksternog radnog takta Pentiuma. Revizija PCI 2.0 predviđa podršku za 64-bitno adresiranje i 3.3 V implementaciju i prvi put definiše PCI konektorske slotove za dodatne kartice. Pomenute PCI specifikacije će procvetati tek krajem godine, kad VLSI tehnologija postane drugi izvor za Pentium PCI čipsal.

Međutim, ukoliko PCI ne postane lektor i na tržištu 486 sistema, mogao bi doživeti istu sudbinu kao EISA serverska arhitekture sa ograničenim dometom. Drugim rečima, ako se PCI suviše uska veže za Pentium i ako broj Pentiuma ostane mali u odnosu na 486. Iržište za PCI dodatne uređaje će biti suviše malo da bi privuklo proizvođače sistema. Kažu da bi osim „Intel"-a bio potreban još neko ko će obezbeđivati PCI podršku za 486, ali se toga još niko nije prihvatio.

Alternativa za PCI je VL-Bus, sabirnica koja je dobila široku pomenu u svetu 486-tice. Firme „Opti" pravi čip-set koji podržava Pentium i VL-Bus i to bi se moglo pokazati kao jeftinije rešenje od PCI sabirnice zbog velikog broja jeftinih VL-Bus kontrolera koji se već nude na tržištu.

Kao i PCI, VI-Bus trenutno radi na 32 bita, ali će specifikacija 1.1 koja se uskoro očekuje podržavati 64 bita. Vakas Kan, direktor marketinga u „Opti"-ju kaže da su započeli VI Bus podršku za Pentium upravo zato što već postoje kontrolen tako da se mašine mogu ponuditi kupcima odmah. Sa druge strane trenutno vlada velika nestešćs kontrolera i dodatnih uređaja za PCI, mede mnogi treba da se pojave tokom sledećih meseci.

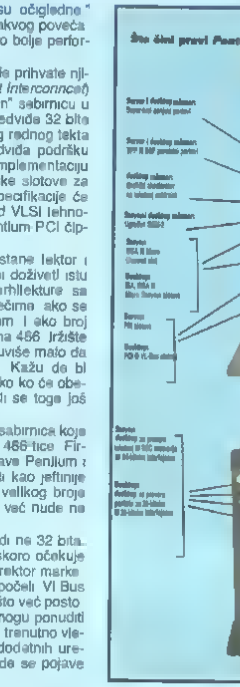
Osnovna zamerka koja se može uputiti VL-Bus sabirnici je u tome što nije tako robusne kao PCI i pri radnom taktu od 50 MHz može „ostati bez goriva", budući da nije predviđena za frekvencije više od 40 MHz. Pošto je VI Bus sabirnice tako usko vezane za 486, potrebne je vezna logika da bi se ona koristila kao „mezanin" sabirnica za Pentium.

Za sponje funkcije, kao što su serijski ulaz/izlaz sa mišem ili modemom, sasvim je dovoljne jeftine ISA sabirnice koje nudi veći izbor opcionih kartica od novih sabirnica. Sa druge strane, dizajneri bi mogli potpuno eliminisati ISA sabirnicu iako što bi konstili samo VL-Bus ili PCI i na nju priključili jednu od novih I/O shema za rad se sponjim uređajima. PCI 2.0 specifikacije predviđa i zajednički slot, teko da možete ugraditi ili ISA ili PCI karticu.

EISA je već druge priče. Predstavnik Micronics-a veruje da će tipičan Pentium server sve-kakako imati EISA slot. „Serveru je potrebno osam ili više slotova a VL-Bus ili PCI ne mogu imati toliko," kaže on, dodajući da su kupci renije zbog cene zaobilazili EISA sabirnice, ali se u skupljim Pentium sistemima one postele relativno jeftinije.

Ali, ime i drugačijih mišljenje. Kari Amdal iz firme „NetFrame Systems" kaže da EISA ne odgovara sistemima za kritične primene jer joj nedostaju skalabilnost i podrška za proveru panteta. Za razliku od toga, NetFrame može za proveru panteta da koristi neograničen broj 25-Mbps MPSA (*Multi processor Parallel Server Architecture*) sabirnica u jednom sistemu.

Firme „Tricord Systems" ime iste zamerke



pa zato konsti vlastitu sabirnicu da bi postigla bolje performanse i pouzdanost procesor memonja i diskovi su na posebnoj, veome brzoj među-vezi (*interconnection*) dok se dodatne kartice kao što je mrežni interfejs priključuju na EISA slot. Isto tako „Parellen" konsti brzu internu sabirnicu za mikroprocesor memonju i SCSI diskove i nudi dvostruke Micro Channel slotove za mrežne interfejse WAN (*wide are network*) veze faks-modem i atmnere.

INTEGRISANJE MATIČNE PLOČE

Pentium sistemi će imati matične ploče se još većim stepenom integracije od 486 mašina. „Što je veća integrisanost, to je niže cene a veća pouzdanost," ističe Tom Mejz, zamenik predsednika NCR-ove grupe u Dejtonu, Ohajo. On predviđa da će za 12 do 18 meseci konsnici dobiti računare sa jednom pločom se procesorom 486 ili Pentiumom, se SCSI-2 DPS i LAN interfejsima na ploči. Neke od ovih komponenti će se pojaviti i ranije u prvim Pentium sistemima.

SCSI je nezaobilazan dao svakog sistema visokih performansi, dok IDE jednostavno ne dolazi u obzir. IDE je usko vezen za ISA sabirnicu i odgovara malom broju lokalnih uređaja, dok SCSI podrževe više uređaje na većim udaljenostima i pruža više fleksibilnosti. Kod servera a možda i na nekim desktop računarima jednosiruki i dvostruki SCSI kontroleri će biti vezeni za lokalnu sabirnicu čime će se postići najveća moguće propusne moć. Sporiji uređaji kao što su strimeri i CD-ROM bi se mogli priključiti na drugi SCSI port na EISA ili Micro Channel sabirnici.

Situacija sa mrežama je nešto manje izvesna, s obzirom na mnoštvo postojećih standarda. Predstavnik firme „Zenith" smatra da integrisanje mreznog interfejsa ima smisla kod desktop ili prenosnih računara ali ne i kod servera. Pošlo proizvođač ne zna za kakvu mrežu će server služiti ostavljaju se otvoreni slotovi. Ali, možda će cak i Ethernet kartica na ploči, koja je do sada imala velikog uspeha, postati suviše ograničena kada se pojave nove mrežne sheme kao što su Fast Ethernet i FDDI.

Najveće šanse za integrisanje u skupe Pentium sisteme ime multimedia. „Intel" neumorno promoviše ugrađnju zvučnog čipa AD 1848 („Analog Device") u Pentium sisteme – reći se o uređaju koji vec koriste „Compaq"-ov *Business Audio* i „Microsoft"-ov *Sound System*. AD 1848 nudi bazični interfejs između analognih signala iz stvarnog sveta i kompjuterskog digitalnog formata (drugim rečima, obavlja 12-bitnu A/D i D/A konverziju brzinom koja dostiže i 44.1kHz CD kvalitete). Međutim konsnička funkcije kao što su glasovno obeležavanje dokumenala ili vezivanje govornih poruka za e-*mail* se moraju obavljati softverski, dok muzičke funkcije zahtevaju dodatni hardver – ili posvećeni smtisažerski čip ili programebilan DSP.

Predstavnik „Micronics" e kaže da još nisu spremni da zaplovi u vode Multimedie. „Ne želimo da stavljamo na matičnu plocu bilo šta što će za godinu dana biti zastarelo," kaže on, „a zvuk i video još uvek prolaze kroz suviše mnogo promena". Dok osnovni delovi kao što je AD 1848 neće zastareti, funkcije koje obavljaju još uvek prolaze kroz evoluciju. Na primer, nezadovoljstvo kvalitetom FM smtetsane muzike vodi ka prihvatanju wave-table audia, dok se stalno javljeju novi video standardi. Gledajući ne duge staze ove funkcije bi mogao preuzeti DSP zajedno sa softverom namenjenim specijalnim multimedijalnim zadacima. Proizvođači lakvih sistema smatraju da potražnja još uvek ne opravdave dodatne troškova koje iziskuje ugradnja DSP a.

Verovatno će na ploči biti integrisani i grafički akceleratori, ali samo za desktop sisteme gde slono izdavaštvo CAD i druge izrazito grafičke aplikacije traže dodatnu snagu. Inteligentni grafički kontroleri (tzv. *bit bitters*) će verovatno biti vezeni za lokalnu sabirnicu kao što su i sada u mnogim 486 mašinama. Predstavnik kompanije „S3" predvida da će tipičan grafički podsistem prvih Pentium masina imati 2 do 4 MB VRAM visokih performansi a ne jeftinji DRAM, pošto o takvim razlikeme u cani proizvođači neće moreti da vode računa. Sa 4 MB video memonje sistem postiže rezoluciju od 1024×768 i 24-bitnu punu paletu boja.

SLEDEĆI VELIKI KORAK

Godinama u pozadini zbog nedostatke standarde i softverske podrške SMP (*symmetric multiprocessing* – simetrični multiprocesing) predstavlja tehnologiju koju mnogi nazivaju „sledećim velikim korakom". Sa pojavom Penjiuma, te operativnih sistema *Microsoft NT* i „Sun Soft"-ovog *Solaris 2.0* mozda možemo konačno reći da je kucnuo čas i za SMP.

Džordž Vajt, predsednik kompanije „Corollary" čovek koji se odevno zdušno zalaže za multiprocesing, kaže da će Pentium naterati RISC obožavaoce da se ponovo vrate Intel-u. „Očekujemo nagli interes za multiprocesing, čemu će doprineti korisnici koji traže pre sve-

NOVA ARHITEKTURA MEMORIJE ZA BOLJE PERFORMANSE

Interfejs za glavnu memoriju jedno je od uskih grla sistema koje se manifestuje pri radu super brzih procesora kakav je *Pentium*. Taj interfejs predstavlja najveću stazu u celom računaru jer se preko njega neprestano prenose programska naredbe i podaci između memorijskih čipova i glavnog mikroprocesora (CPU). Ako memorija ili staza ne uspevaju da održe korak sa stalnim zahtevima glavnog mikroprocesora CPU je prisiljen da čeka i tako se gubi dragoceno procesorsko vreme.

Današnji DRAM čipovi – poznati i kao asinhroni *page-mode* i generički – su ograničeni keko svojom unutrašnjom arhitekturom tako i interfejsom za memorijsku sabirnicu glavnog procesora. DRAM arhitekture se nije značajnije menjala još od 1974. kao ni memorijski interfejs u PC računaru osim što su se memorijske sabirnice proširile – od 8 bita kod procesora 8088 do 64 bita kod *Pentiuma*. Iako su šira sabirnice povećale propusni opseg sirovih podataka on još uvek zaostaje za vrtoglavom brzinom kojom rade brzi mikroprocesori.

Sve češće PC računari pramoćuju taj raskorak pomoću brzih SRAM (statični RAM) čipova, koji keširaju sabirnicu između glavnog procesora i DRAM-a (tipičan 486 ili *Pentium* računar može imati 256K SRAM keša. Ali SRAM je mnogo skuplji nego DRAM a povećanje keša preko 256 KB dalja manju učestalost vraćanje (*rate of return*).

Da bi se prevazišle ova ograničenja razvijeno je nekoliko novih tehnologija. Većina zahteva novi tip DRAM-a dok su dva usmerena ka rešavanju problema memorijskog interfejsa. Za sada se još uvek ne može reći koja od ovih tehnologije će postati DRAM standard. Sve su ze oko 15% skuplje od generičkog DRAM-a ali ipak mogu smanjiti ukupnu cenu sistema jer alimiraju SRAM keš i prateće kontrolere. Ovo bi moglo doprinati i smanjenju matične ploče koja će trošiti manje struje što je značajno za prenosiva sistema.

EDRAM (enhanced DRAM)

EDRAM (enhanced DRAM) je nastao u kompaniji „Ramtron" (Kolorado) i to je jedini DRAM koji se isporučuje u velikim količinema. Njegov je pristup evolucioni i integriše mali SRAM keš se brzim jezgrom generičkog DRAM-a. Svaki EDRAM čip ima 2 Kb SRAM a od 15ns (nanosekundi) i 4 Mb DRAM a od 35-ns.

„Ramtron"-ovi *benchmark* testovi pokazuju da mašine sa EDRAM om daje bolje performanse od sličnih sistema sa generičkim DRAM-om i SRAM keš om osim u slučaju da aplikacioni program potpuno staje u keš kaoa su performanse iste kao kod drugih sistema. „Ramtion" ime već 44 kupca EDRAM a je ugrađuju se u razne mašine – od desktop računara preko radnih stanica, do laserskih štampača i kopi aparata. Međutim ni jedan od prvih *Pentium* sistema nema EDRAM.

CDRAM (Cache DRAM)

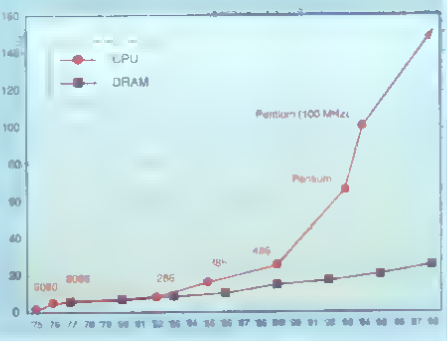
CDRAM (Cache DRAM) nastao u „Mitsubishi"-ju je sličan EDRAM u. Integriše SRAM keš sa 4 ili 16 Mb DRAM e. Iako je keš CDRAM-a ne ploči veći (16 Kb naspram 2Kb) DRAM je aporiji (70ns nasram 35ns) SRAM koji CDRAM ima na ploči može poslužiti kao keš ili kao bafer zavisno od toga da li aplikacije zahteve serijsko ili direkino pristupanje podacima.

Kada se podaci dobijaju serijski – npr za osvežavanje bi mapiranog ekrana – CDRAM može unapred pranati podatke iz svog DRAM jezgra u SRAM bafer i time povećati performanse. U stvari „Mitsubishi" tvrdi da CDRAM koji ima jedan port, takve operacije odavija brže od VRAM a (video RAM) koji ima dva porta. PC računar se CDRAM-om će po brzini biti sevan sličnoj mašini se DRAM-om i sekundamim SRAM kešom od 256KB.

Za sada CDRAM isporučuje samo „Mitsubishi" ali bi uskoro trebalo da ga proizvode NEC i još jedna kompanija.

SDRAM (Sinhroni DRAM)

Raskorak u performansama između DRAM-a i mikroprocesora



SDRAM (sinhroni DRAM) je takođe evoluciona alternativa a najšira je najveću podršku proizvođača poluprovodnika. Krajem ove godine SDRAM čipove će početi da isporučuju „Mitsubishi" NEC „Samsung" „Taxas Instruments" i gotova svaka ko nešto znači na DRAM tržištu. Da bi SRAM čipovi bili međusobno zamenjivi radi se na uspostavljanju standarda.

Za razliku od današnjih asinhronih DRAM ova SDRAM razmenjuja podatke se glavnim procesorom sinhronizovano sa signalom eksternog časovnika i tako dostiže punu brzinu CPU/memorijske sabirnice ne uprosvajući procesor. Na primer 16-Mb SDRAM (Texas Instruments) čiji se prvi uzorci očekuju krajem godine može da razvije brzinu do 100 MHz. To je više nego dovoljno za *Pentium* a ime prostora i za brza procesore.

SDRAM najbolje performanse daje pri serijskom transferu podataka. „Texas Instruments" leliče da je idealan za aplikacije kao što su tekst-procesori radne tabele i multimedija. Međutim kod programa koji se mnogo oslanjaju na direktan pristup (npr baza podataka) memorija se kešom kao CDRAM ili EDRAM će varovatno dati bolje rezultate.

Rambus DRAM

RDRAM (*Rambus DRAM*) proizvod firme „Rambus" ne revolucionaran način pristupa problemu propusnog opsega memorije. Osim što je uveo novi tip mamorijskog čipa, „Rambus" je smislio i novi interfejs za glavni procesor.

RDRAM čipovi su vertikalni i sve su im nožice na jednoj strani. Razmenjuju podatke sa glavnim procesorom preko 28 vodova koji nisu duži od 12 centimatara. Sabirnica može adresirati do 320 RDRAM čipova i predviđene je za brzinu do 500 MBps mada 400 do 450 više odgovara realnosti.

RDRAM čipovi nemaju SRAM-a ne ploči ali se stranice keširaju pomoću senzorskih pojačala. Kontroler koristi novi tip I/O ćelije a sabirnici nije potrebna vazna logika. Ovi čipovi se mogu proizvoditi u postrojenjima u kojima sa proizvodi generički DRAM.

„Rambus" trenutno isporučuje uzorke RDRAM a od 4-Mb svim vodećim proizvođačima sistema dok masovna proizvodnja treba da počne na jesen. U pripremi je i RDRAM od 16 Mb a licencu za „Rambus" tehnologiju su već okupili „Fujitsu", „Hitachi" NEC i „Toshiba".

RamLink

RamLink predstavlja najrevolucionarniju tehnologiju od svih ali je predviđena za nešto daiju budućnost. Umesto je na interfeje između glavnog procesora i memorije e ne na internu arhitekturu memorijskog čipa. *RamLink* je začet u IEEE ali su u njegovom razvoj učestvovale mnoge firme uključujući „Apple" „Hewlett-Packard" TI kao i najveće proizvođače DRAM-a.

Ova tehnologija proizlazi iz IEEE standarda poznatog pod imenom SCI (*Scalable Coherent Interface*) kojim se definiše sistemska arhitektura koja se svuda uklapa, od jednog do 64 000 procesora. *RamLink* je memorijski interfejs se prstenasto raspoređenim vezama kao mreža. Saobraćajem na prstenu upravlja memonjski kontroler. On šalje poruke DRAM čipovima, koji služe kao nodovi. Ostali nodovi mogu biti ROM čipovi fleš ROM drajvovi pa čak i drugi *RamLink* prstenovi.

Hans Vigera koji radi za „HP Labs" i predsedavajući je IEEE *RamLink* komisije leliče da *RamLink* može postići brzinu od 500 MHz pa čak i od 1 GHz. Ali treba da prođu godine da *RamLink* postane realnost.

Ne čak i ako za seda ostavimo po strani *RamLink* i okrenemo se tehnologijama koje se bore za svoja mesta u skorijoj budućnosti nemoguće je predvideti koja će od njih – EDRAM CDRAM SDRAM ili RDRAM – biti usvojena kao novi standard.

Proizvođači sistema nisu spremni da prihvate ni jednu od alternativa sve dok se nedvosmisleno ne pokaža koja je najbolja. Niko na želi da napravi računar sa RAM-om koji je ne tržištu i odak i kasnije ga je teško dograditi.

Za sada su nove memorije za oko 15% skuplje od postojećih ali se taj odnos brzo može promeniti. Sve zavisi od broja prodatih primeraka – denas su skupi jer ih niko ne kupuje.

Neve memorije bi mogle postepeno prodreti ne tržište popunjavajući posebne segmente tržišta. Nude velike prednosti za izvesne ugrađene (*embedded*) aplikacije i mogu zamenti VRAM na video karticama. Sa porastom prodaje i smanjenjem cena, mogli bi polako zauzeti mesto glavne memorije.

Druga mogućnost je da će evolucioni dizajn kakav je SDRAM popuniti potrebu za bržim memorijom u sistemime iz *Pentium* klase. Kasnije kada brzina procesora dostigne 1 GHz, revolucionarne rešenje kao što je RDRAM ili *RamLink* bi mogle donesi spas od novog uskog grla.

Ali vrlo je verovatno da nijedno rešenje neće prevladati. Jedan analitičar leliče da je svetsko tržište memorija prošle godine vredelo ti 5 milijardi dolara što znači da ima dovoljno prostora za više od jedna arhitekture.

---

ga bolje performanse te će se zato oslobodili predrasude koje su gajili prema Intelu.

Neki se, opet, ne slažu se predstavnikom „Corrolary"-ja i smatraju da SMP neće biti op- šteprihvaćen sve dok ga ne podrže najveći proizvođači čip-setova. Međutim, multiproce-

sna arhitektura koju konste superserveri od „Compaq"-ovog *Systempro* do „Parallan" i „Tricord" sistema, uglavnom je vlastite proizvodnje i u mnogim slučajevima nije simetrične SMP bi mogao prodreti ne šire tržište preko čipa C-Bus II za koji „Corollary" daje licencu i koji podržava *Windows NT*.

Međutim, sam „Intel" je još više zakomplikovao situaciju kada se proširile glasine o P54C, nekoj vrsti *overdrive*-a za Pentium, koji umesto da onesposobi postojeći čip (kao OverDrive za 486) radi uporedo s njim čime se postiže multiprocesing. „Intel" će ovo ponuditi kao rešenje za brzo i lako unapređenje performansi ne desktop računarima a ne kad skalabilnu multiprocesnu arhitekturu.

„Ljudi se pribojavaju da bi ovakav pristup mogao stvoriti loš glas o multiprocesingu se „Intelovim" mikroprocesorima" kaže Vejl P54C od proizvođača sistema zahteve samo dodavanje jedne uticnice ali na doprinosi porastu performansi osim kod „multithreaded" programa u okruženjima kao što su NT ili OS/2. To nije pravo SMP rešenje, jer čipovi koriste istu sabirnicu i eksterni keš.

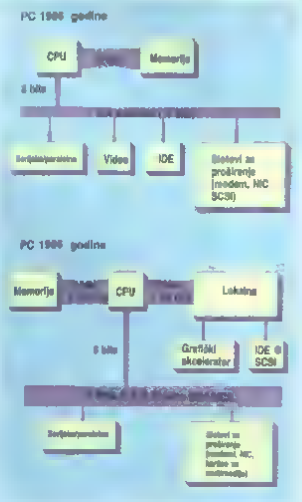
Drugu vrstu multiprocesinga može pružiti redundantni dizajn za bolju toleranciju grešaka. Predstavnik „Opti"-je Khan, kaže da je ova tehnologija poznata već godinama, ali do sada nije primenjivana zbog ljute borbe za niže cene. Sa Pentium-om su svoju priču platformu dobili korisnici koji traže pouzdan multiprocesing se dvostrukim procesorom."

Predstavnik kompanije „Zenith" predviđe da će Pentium sistemi preuzeti na seba host funkcije koje nisu bile moguće na PC računarima – obavljanje dijagnostike sa velikih udaljenosti automatski *dial-out* upravljanje mrežom na nivou sistema.

Istog je mišljenja i Mallos iz HP-a. „Kupci traže sve više i više funkcija. To vodi ka primeni redundancije ECC memorije (*error checking and correction* – provera i ispravljanje grešaka) štimera i dijagnostike a upravo to su stvari koje polako postaju osnove sistemske arhitekture.

Ne tom polju prednjače proizvođači superservera. Na primer „Peralian" već ima ECC memoriju i paritet na svim sabirnicama, i nudi sofisticirano praćenje i dijagnozu. Njihov predstavnik tvrdi da je održavanje „Parallen" sistema rađeno bolje nego na *mainframe* računarima, jer se koristi grafički korisnički interfejs (GUI). Sledeća verzija softvera za održavanje sistema će imati podršku za SNMP protokol tako da će „Peralien" serveri moći da rade u heterogenim sistemima za upravljanje mrežom „Tricord" i „NetFrame" serveri takođe nude visok nivo integriteta podataka i dijagnoze.

Tome će doprineti i sam Pentium koji ima nekoliko novih mogućnosti dijagnostike. Ne čipu se nalaze registri za praćenje performansi, koji beleže statističke podatke o postotku pogodaka keša, transakcijama preko sabirnice i prosečnom čekanju procesora. U skladu sa takozvanim *Joint Testability and Automation Group* standardom ovi interni registri se mogu direktno čitati što bi trebalo da pomogne kod otklanjanja bagova i testiranje sistema. Osim toga, Pentium-ov režim FRC (*functional redundancy-checking* – funkcionalna provera redundanse) omogućava da dva čipa rade uporedo i da se njihovi rezultati porede u stvarnom vremenu. Na žalost, FRC režim u slučaju greške ne daje podatak o rezolucji, tako da služi za otkrivanje ali ne i rešenje problema.



Trend u dizajnu PC računara je premeštanje funkcija sa ISA sabirnice koja služi za opštu namenu na šire, specijalizovane sabirnice vezane za brzinu mikroprocesora (CPU) 8-bitni interfejs ISA sabirnice i radni takt od 8 MHz čine uske grle za sve periferne funkcije. Zato današnji sistemi imaju 32-bitne ili 64-bitne staze od mikroprocesora do memorije i 32-bitne ili 64-bitne lokalne sabirnice za brze podsisteme, kao što su video i skladištenje podataka. I dalje se koristi ISA sabirnica za sporiji ulaz/izlaz i za održavanje kompatibilnosti sa starijim sistemima.

INVESTIRANJE U PENTIUM

Prva generacija PC računara se Pentiumom još uvek ne koristi sve prednosti novog mikroprocesora. Onaj ko je čvrsto rešio da već sada kupi Pentrum mašinu trebalo bi barem da se potrudi da dobije PCI ili VL-Bus za video ili disk I/O 64-bitni mamonjski interfejs i 32-bitna I/O sabirnica će biti lep bonus.

Međutim, bolje bi bilo da se sačeka da ostatak arhitekture uhvati korak se čipom te da prorade operativni sistemi koji u potpunosti konste prednosti Pentiuma. Ukoliko niste sasvim sigurni za šta ćete koristiti računar a planirate da ipak nabavite 486 lepo bi bilo da to bude mašina sa P24T utičnicom na ploči (za „Intelov" još nenajavljeni Pentrum *overdrive* procesor), s tim da se pomirite s tim da, kada se pojavi odgovarajući čip nećete dobiti iste performense kao originalni 64-bitni Pentium sistem.

Izvor *Byte*

Prevela *Ranka Jovanović*

POVRATAK SKROMNIM I/O PORTOVIMA

Sistem ima munjevit procesor i arhitekturu koje odgovara njegovoj snazi: ali sa pri štampanju dokumenta podaci zaglevljuju na prolasku kroz paralelni port koji propušte samo 150 KB u sekundi. Tu oči gledno nešto nije u redu.

Se dolaskom *Pentium*-a proizvođači sistema su počeli ozbiljno da preispituju paralelne i serijske I/O portove. Oni sada podnose sve veća opterećenja – ne samo za štampanje već i za druge uređaje od strimera do mrežnih interfejsa – i postaju usko grlo za performanse sistema.

Mnogi proizvođači su počeli da primenjuju nova tehnologije porta koje dramatično ubrzavaju izlaz odnosno ulaz. Novi portovi su ujedno i pametniji nude nove sposobnosti i lako smanjuju ternl konfiguracije. Neki od njih će se ugrađeni u sisteme, naći u prodaji već ove godine.

Brži paralelni port

Unapređenje standardnog paralelnog porta je počelo pre nekoliko godina kada je „Xircom" prezvodač mrežnog interfejsa za paralelni port, pomoću „Intel"-a da definiše brzu specifikaciju *Enhanced Parallel Port* ili EPP je bio podržan u „Intel"-ovom čip-setu 386SL i postao je sastavni deo *notebook* računara, podigavši propusni opseg se 150 KB na 2 MB u sekundi.

I „Hewlett Packard" je bio zainteresovan za ubrzanje paralelnog porta zbog performansi štampače LaserJet. Ie je razvio tehnologiju nazvanu „Zippy" koja postiže brzinu od 2-5 MB u sekundi. *Zippy* je dvosmeran što znači da štampač izveštava računar o svom statusu ili o pojavi greške. Nekon što je „Microsoft" udružio snage sa HP-om *Zippy* je prilmenoven u *Extended Capabilities Port* ili ECP. Nedavno je IEEE komisije koja se bavi usavršavanjem paralelnog porta odlučila da sa EPP i ECP spoje u jedan standard 1284.

Očekuje se da proizvođači čip-setove i BIOS e kao i HP i drugi proizvođači štampača i skenera u najnovijim proizvodima imaju podršku za IEEE 1284. Povećanje cene sistema bi trebalo da bude zanemarivo. „Microsoft" će u sledećoj verziji *Windows*-a dodati podršku za dvosmernost – tako bile u dijalogprozoru *Print Manager*-a mogli naći poruku da štampaču nedostaju papir ili toner.

Šanse u serijskom portu

Inženjeri još veća šanse za unapređenje vide u serijskom I/O portu nego u paralelnom zauzima manje prostora zadaje manje problema se elektnkom (npr *cross-talk*– preslušavanje) i lakše se prilagođa va optičkim medijima. Tokom sledećih god ne može se očekivati primena dve nove tehnologije *Access bus* (Philips Semiconductors) i *Serial Bus* (IEEE).

Access.bus liči na ADB (*Apple Desktop Bus*) ali je mnogo bolji – brži je i podržave više uređaje Tehnologiju je razvila firma ČMDNMČ"Signetics" a omogućava priključenje preko 100 serijskih uređaja – testatura miševe trekbola digitalizatora – na je dan 100-KBps port što eliminiše potrebu za više ko nektora na poleđini sistema. Uređaji se sam identifikuju i mogu se priključiti i dok sistem radi (*hot-plugging*) DEC već koristi *Access bus* u radnoj stanici *Maxime* dok je „Logitech" proizvođač odgovarajućeg miša.

Još bolje performanse pruža *Serial Bus*, poznat kao IEEE P1394. Sada ime propusnu moć od 100 MBps ali bi mogao doći i 400 Mbps. Se takvom brzinom očigledno je da nije predviđen da bude sa mo port za miša. U stvari *Serial Bus* je zamišjen kao interfejs opšte namene koji može zameniti čitav niz I/O tipova, uključujući i ADB RS-232 RS-422 paralelni i SCSI. Po rečima Majkla Tinera inženjera Apple a i bivšeg člana komisije za P1394 „biće to standardna I/O veza čija će upotreba prevazići Apple pa i računara."

Kao i *Access.bus*, *Serial Bus* zahteve samo je dan jeftini konektor na poleđini sistema priključenim uređajima nisu potrebni DIP prekidači niti džemperi – oni se sami identifikuju i konfigurišu. Za razliku od SCSI niz *Serial Bus* uređaja neće zahtevati adrese ni terminatore ali će performanse meći da se mera sa onim koje pruža SCSI-2.

Članovi grupe IEEE 1394 su „Apple" IBM i NCR. Priča se da je „Apple" već razvio *Serial Bus* kontrolere čiju će licencu prodavati drugim kompanijama. Tiner potvrđuje da su završeni hardverski i softverski prototipovi i očekuje da će „pravi" hardver ugledati svetlost dana 1994. Idealni kandidati za ugradnju *Serial Bus* porta su *Pentium* i *PowerPC* sistemi.

PERFEKCIJOM DO SAVRŠENSTVA
Design by CRNO NA BELO
BS PROCESOR
BEOGRAD Hadži Nikole Živkovica 2
telefoni 011/162 230 181-311 lok. 404-405
telefax: 011/180-739
NIŠ telefoni 018/51-000 331-854
NOVI BEČEJ telefon. 023/772-153

BILO KUDA WINDOWS SVUDA

Ako je varovati Microaoftu, naki članovi raatuča porodice *Windowa* operativnih sistema kontroiisaće ave, od prijemniks za kablovaku televiziju preko vašeg atonog PC računara, vaša računaraka mreža, do višeprocaaorakog RISC superservera u vašoj firmi. Pitanje koje Microaoft želi da postavite nije „Da li da koriatim *Windows*?", već „Koji *Windows* da koriatim?".

Hor glasova koji postavljaju sve pitanje ave je glasniji nakon pojave *Windows NT* sistema. Napredne osobine *NT*-a su sada već dobro poznate mikrokernel arhitektura prenosivost izvršavanje niti na simetričnom višeprocesor skom sistemu objektno orijentisana zaštita podataka i ugrađene osobine više različitih operativnih sistema.

U ovom članku odgovoriću na pitanje koje Microsoft želi da vi postavite kola je od 16- i 32-bitnih verzije *Windows*-a najbolje za pojedine primene? Međutim odgovoriću i na pitanja koje Microsoft ne želi da se postavlja u kojem slučaju *Unix* OS/2 NetWara ili Macintosh mogu biti bolje rešenje da *Windows*-a?

Grafik performansi *NT*-a ima graciozan luk preko Pentiuma i R4000 do DEC Alpha AXP najbržeg RISC procesora na svetu i dalje u svet SMP

Sistem se dva procesora neće automatski biti dva puta brži od jednoprocesorske mašine iz ietog razloga iz kojeg 32-bitni program neće biti duplo brži od 16-bitnog Performanse se ne poboljšavaju linearno osim ako se linearno ne povećava propusna moć procesora a procesori već sada ostavljaju I/O podsisteme daleko iza sebe. Na sreću jasna razdvojenost *NT*-a od harduera mu kome se izvršava ohrabruje inovativne projekte koji treba da prevaziđu to ograničenje.

SVE U *Win32* PORODICI

Windows programe sam pisao da budu prenosivi sve od pojave *Windows-a 3.1* U početku je to značilo da se isti izvorni kod mogao prevesti na jedan način za *Windows 3.1*. I na drugi način za *Windows NT* Onda se pojavio treći cifjru sistem *Win32s* Kao i proširenja *Windows*-a koja nude Watcom i Rational Systems *Win32s* daje *Windows* programma ne *3.1* kernelu pristup linearnoj 32-bitnoj memoriji Za razliku od drugih proširenja, *Win32s* izvršava binarni kod prevođen za *Windows NT Win32s* nema napredne osobine *NT*-a – niti zaštitu podataka, fleksijeove krive – koje nedostaju i u DOS-u i *Windows-u 3.1* Zapravo osim *Win32* API funkcije za upravljanje memorijem, *Win32s* nema ni jednu od naprednih osobina.

Unatoč svom nedostatku, zbog kojeg se Microsoft izvinjava čak i imenom („s" u imenu označava *subset* ili podskup) *Win32s* će ove godine naći mnogo više korisnika nego *Windows NT* Da bi videli zašto, pogledajmo ACIS – program za 3-D modeliranje i renderovanje firme „Spatial Technology" (Boulder Colorado) koji je nastao u 32-bitnom *Unix* okruženju „Spatial" razvija *Windows* verziju ACIS-a za *Windows NT* ali Dick Sowar osnivač kompanije ima jasnu sliku o tome gde je njihovo tržište Imaćeme nekakvu prodaju ne *NT* tržištu ali naš primarni cilj je *Windows 3.1*" kaže on „Mnogo je brže izvršavanje ne *32s* nego pod *NT*-om na ietoj mašini I to je okruženje koje ne raspolaganju ima više od deset miliona korisnika *Windows-a 3.1* Počećemo da isporučujemo ACIS za *32s* čim an bude raspoloživ" „MathSoft" (Cambridge Massachussets) i njihov *Mathcad* biće još jedan *Win32s* proizvođačnik.

Kome je uopšte potreban *NT*? Pogledajmo još jednu *Unix* aplikaciju koja se proizvodi za *NT*, *Pro/Engineer* firme „Parametric Technology" (Waltham, Massachusetts) *Win32s* stvarno obezbeđuja linearnu memoriju kaže Mike Payne potpredsednik za razvoj, „ali da bi radio se 17MB našeg koda potreban je pravi sistem za upravljanje virtualnom memorijom, a ne beda koju možete naći u *Windows-u 3.1*" Payne kaže da se *Win32s* verzije *Pro/Engineera* dobro prenaše pod beta verzijom *NT*-a na Intel r R4000 hardveru i očekuje da će preuzeti deo tržišta Hewlet-Packard Silicon Graphics i Sun verzija programa.

Porad boljeg upravljanja virtualnom memorijo *NT* ima više dobrih strana. Nakon što sam ga koristio više mesaci zaključio sam da se radi o najrobusnijem sistemu nakon VMS-a. Ta nije slučajne, pošto je vodači arhitekta *NT*-a David Cutler radio ne razvoju VMS-a kod DEC-a, a zatim preveo svoj tim kod Microsoft-a. Kompanija kojoj život zavisi da jedne aplikacije verovatne ne bi trebale da je *gradi* ne DOS/*Windows* temelju Izbor tada postaje *Unix* OS/2 ili *NT* Svaki od ovih sistema ima develjno prepiruka da može završiti posao.

Draž *NT*-a je kompatibilnost sa *Windows-om 3.1* ne nivou izvornog i mašinskog koda. *NT* bolje radi sa *Windows* kodom jer direktno prevodi *Windows* API poziva u *Win32* podsistem To znači da se 16-bitni kod dobar deo vremena izvršava u pravom *Win32* režimu, ne RISC ili CISC platformi Ako odlučite da konvertujete *Windows 3.1* program i ako želite da poboljšati neka jednostavna uputstva (za precizniji) prevešćete više vremena pišući novi *makefile* nego menjajući sam tekst programa.

PC U RADNOJ STANICI

Iake je otvoreno pitanje da li je *Windows* imalo bolji GUI od *X/Motif*-a ili OS/2 *Presentation Manager*-a, sigurna je da *Windows* aplikacija ne može *Windows* platformu kao što je *NT* nudi jasna prednosti Kompanije koje koriste radne stanice primećuju da pored tih stanica ničiu PC računari Zašto? Kada ne trguju akcijama, ili kada ne rade analizu konačnih elemenata, ti ljudi žele da pišu pisma i koriste programe za rokovnike i agende – i čini se da to žele da rade pod *Windows*-om

Ovo je jedan od razloga zašto „Intergraph" (Huntsville Alabama) prenosi *NT* ne *Clipper* i računare svoju liniju RISC radnih stanica, kae i zašto prenosi kompletnu liniju CAD/CAM aplikacija se svog *Unix* sistema ne *Windows*. „Interesantno je da se rešenje za uspešnavanje funkcija radne stanice i PC računara u jednu kutiju pojavljuje kao korak naniže *Windows*a umesto kao korak nevise *Unix*", kaže Jim Ruester, direktor za odnose se javnošću

Prelazak ne *NT* značiće još nekoliko stvari za petog svetskog proizvođača radnih stanica. *NT* će otvoriti *Intergraph Clipper* koji je do sada bio označen kao zatvorena i sasi svojstvena RISC arhitektura. Sličiba takođe drastično pojednostavljuje „Intergraph"-ovu meiju prenošenja softvera ne Intel i alternativne RISC platforme Povuđena jednolikost *NT*-a znači da „Intergraph" može da prikaži skup platformi na kojima radi bez potrebe da luta kroz šumu različitih *Unix* API-ja.

Štaviše, bogata *Windows* okruženje biće plodne se za „Intergraph" softver „Kada postavite inženjerske softver zajedno se poslovnim aplikacijama i sistema za iš-

DOĐITE U Epicentar

13.09. - 18.09.93.
OD 10 DO 18 SATI
DOM OMLADINE BEOGRADA

MICROSYS COMPUTER SHOW

Svaki posetilac MICROSYS Computer Show je potencijalni dobitnik neke od vrednih nagrada. Izvlačenje nagrada biće sprovedeno 18.09.93. na samoj izložbi.

INFORMACIJE:

MICROSYS Beograd, (011) 432-690, 430-059

MICROSYS Novi Sad, (021) 52-550, 611-366

POŽELI NEŠTO

Popuni ovaj kupon i predaj ga na desku za informacije MICROSYS CompUter Show, Dom Omladine Beograda od 13.09. do 18.09.93.god.

Prezime _____

Ime: _____

Adresa: _____

_____

nu produktivnost može deći do razvoja kerlenih nusprodukata' kaže Tom Steele predsednik „Intergraph"-a za softverske sisteme. „OLE zakačen za radnu tabelu vrlo je zanimljiv za nas". On dodaje da će sa prenos poruka i *peer to-peer* umrežavanje koje *NT* deli sa *Windows for Workgroups*, lepo uklopiti u deo koji se u „Intergraph" aplikacijama bavi tokom rada na projektu.

Chicago I STRAH OD OS/2

Izvan scene koje se još i mlađi poluetar *Windows*-a integrisani DOS/*Windows* sistem po imenu *Chicago* (ili *Windows 4/DOS 7*) koji će izvršavati još jednu varijantu *Windows* API-ja. *Win32c*. Detalji se samo nasluću razumljivo je da Microsoft na žalost da subotora trenutke slave *NT*-a. Međutim izvan ekranaju da je *Chicago* upravo onakav „*NT* Lite" kakav Microsoft poriče da jeste.

Izgleda da će *Win32c* zauzimati mesto na srednji krive koja povezuje *Win32* se *Win32s*, i da će podržavati nti semafore i Bezierjeove krive ali ne i *Unicode*-zaštitu podataka. Žrtvujući prenosivost i SMP opciju ovaj novi 32-bitni *Windows* biće namenjen 80x86 procesorima i Pentiumu u mrežnim konfiguracijama sa 4MB memorije.

Za neke piece softvera koji su se gadno opekli načinom na koji se Microsoft odnosio prema OS/2. *Windows NT* (u svetlu pojave *Chicago* projekta) izaziva uz neminavljući osećaj već viđenog. „OS/2 je bio jeden pravac i razvoja" kaže jeden istaknuti proizvođač softverskih alatki „a onda su načinili *Windows* boljim i ubili OS/2. Sada je *NT* pravac i razvoja a oni evo, opet poboljšavaju *Windows*"

Međutim, drugi posmatrači ističu da je to je na vidiku API koji je potpuniji od *Win32s* i vezan za moćniji *Windows* više klase. Rasprostranjem *Windows*-a preko šireg spektra hardvera, samo će se poboljšati izgledi *NT*-a. „Zagolicalo nas je kada smo čuli za *Chicago*" – kaže jedan pisac softvera „jako i razumemo potrebu Microsoftta da minimiziraju sve te priče jer bi mogle zbuniti ljude u vezi sa *NT*-om"

IZBOR ODGOVARAJUĆE TEHNOLOGIJE

Strategija „*Windows* svuda" ima smisla utoliko što ujedinjuje sisteme u rasponu od laptopa do superservera DOS i *Windows* nastavlće da dominiraju u nižoj klasi. Na radnom stolu međutim granice neće biti tako jasne *Windows 3.1* (sa ili bez *Win32s*) *Chicago* i *NT* će naći svoje delove tržišta, određene hardverskim resursima korisnika i njihovim potrebama.

Na mojem 8-megabajtnom 386/25 koje sa izvršava *Windows for Workgroups* i *Win32s*, i očekujem da će se isto tako fino ponašati i se *Chicagom*, ali računar na bih opterećivao *NT*-om. Moj 486/50 se 16MB memorije, a druge strane jeste lepa *NT* mašina. Konstači ga kao radnu stanicu, dopada mi se njegova stabilnost i dobar odziv u višeprogramskom radu. Takođe ga koristim kao *file server* i platformu za SQL Server.

Na mojoj *NT* mašini sada se izvršava beta-verzija SQL Servera ali sam nekoliko meseci pre nego što se pojavila koristio kupovnu verziju SQL Server 4.2 a to je jedna OS/2 aplikacija – i to vrlo zahtevna. Nikada nisam stvarno očekivao da će OS/2 emulator biti dorastao takvom poslu. Začujuće da jeste pa sam koristeo tu konfiguraciju za testiranje serije SQL Server klijent aplikacija. Teorija mikrokernela kaže da on može podržavati bilo koji napredan operativni sistem na robustan način ako su upravljanje memorijom niti sinhronizacija i ostali servisi korektno implementirani. Moja iskustva se OS/2 podsistemom *NT*-a sugerišu da je teorija korektna, i ukazuju na snagu i fleksibilnost *NT*-a.

Da li je *NT* za radne stanice ili za servere? Ja kažem da je za oboje. Sasvim je jasno da je to serverska platforma, ali je verujem da su moji poslovi isto tako bitični kao i poslovi servera. Prema tome koristeo *NT* (kada hardver dozvoljavaš jer kada *Windows* pali od nekog od svojih nepredvidivih kraheva, je gubim vreme vraćajući sve u radno stanje i ponekad gubim podatke. Očigledno spelli *NT*-a za hardverom – na Intel platformi trebaće vam 12 do 16MB RAM memorije i procesor 486 – onemogućava svaj izbor većini današnjih korisnika.

Čak i ako na mašćete koristili *NT* međutim, možete se odlučiti da se menjate za njegovu stabilnost (koja vas košta potrebnog sistemskog vremena) za brzinu koju možete dobiti uz *Windows*-a, izvršavajući *Win32s* aplikacije na istom hardveru. Peenta nje u tome za koju ćete se opciju odlučiti, već jednostavno u tome da imate mogućnost izbora. Širok izbor *Windows* aplikacija bače na raspolaganju svuda je ih zajedničko API navedi da slobodno migriraju između platformi.

Microsoft tvrdi da će zajednički API olakšati razvoj klijent/server aplikacija. Danas takve aplikacije pokrivaju *Unix* OS/2 i *NetWare* servere i *Windows* klijente. Te prisiljava peca softvera da se rvu sa dva čupava API-ja umesto samo se jednim. „Te nje vrlo važne ako koristite i serverski softver u paketu" kaže Reed iz Oracle-a, primećujući da mnogi TCP/IP proizvodi za *Windows* podržavaju konćenje Oracle servera za *Unix*. „Ako sami razvijate klijent/server aplikacije onda jeste važne – zajedničkr API uštedeće vam muke"

PEN COMPUTING I *Windows*

Kada ste poslednji put videli da neko u svoju koristi sistem se olovkom? Danas ti tipične kancnik takvog sistema mogao da se vidi kako ide kraj auto-puta i proverava električne vodove za detretaku elektrodistribuciju ili kako popisuje zalihe u K-Mart sledeću. Pravi izbor za ovakve pokretne zadatke prilagođenja podataka jeste *GridPad* i *PenDOS*.

U zavetrini čekaju dva elegantna rešenja. *Microsoft Windows for Pens* i *PenPoint* firme „Go Corp" Valjano Microsoftovo proširenje *Windows*-a definiše put evolucije glavnih *Windows* aplikacije od rešenja koja im samo omogućavaju da olovku koriste do onih koje su na olovci i zasnovane. Revolucionarni *PenPoint* firme „Go" zaboravlja na kontinuitet sa desktopom i izmišlja vlastiti supernoviji *notebook* korisničk interfejs. Na nesreću njegan od ovih pristupa ne znači mnogo onom inspektoru detretske elektrodistribucije od malopre čija bi reakcija na Microsoftovu mantru „*Windows* svuda" verovatno bila samo „Šta reče?"

VIŠE PROCESORA, VIŠE NITI

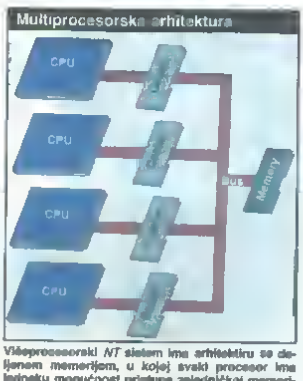
Ono što može učiniti jedan procesor brža će raditi brže a ih još brže. Ta je teorija višeprocesorskih mašina, između teorije i sistema koji rade međutim, leži jaz koji se može premostiti samo pomoću kompleksnih i skupih tehnologija. Deljene memorijske magistrale i protokoli za koherentnost *cache* memorije koriste se godinama i svoj put su našle i u izvesnom broju komercijalno raspoloživih računara, ali do sada višeprocesorski način rada nije imao mnogo ubcaja na tržište.

Windows NT može sve te promeniti. Prema Tamu Hieku, sistemskom inženjeru grupe za multiprocesiranje firme NCR. „Ključni momenat koji omogućava višeprocesorski rad jeste operativni sistem koji te omogućava". NCR koji je u meju vremeno da predstavi liniju višeprocesorskih mašina zasnovanih na Pentiumu očekuje zdravu konkurenciju AST Research, Compaq, Sequent, Siemens Nixdorf i Wyse su među kempanjama koje će predstaviti sisteme se simetričnim multiprocesiranjem zasnovane na *NT*-u. Ove i druge kompanje očekuju da *NT* bude podešcaj za multiprocesorsko tržište.

Jednoprocesorska mašina predstavlja za *NT* degenerisani slučaj iz temelja projektovan za SMP kernel *NT*-a jednostavno alocira niti na onoliko procesora koliko ima na raspolaganju. U mašini se četiri procesora će se uvek istovremeno izvršavati niti se najvišim prioritetom. Kernel vodi evidenciju o njima koje su spremne za izvršavanje zajedno se njihovim prioritetima, i automatski vodi računa o balansiranju opterećenja procesora.

Ovakav pristup razlikuje se od pristupa koji se koristi kod *Unix* višeprocesorskih računara. oćim kod onih zasnovanih na Mach procesoru SCO *Unix* i *Unix* System V release 4.0 postižu višeprocesorski način rada pomoću proširenja kernela. Prema Mike Flanary-ju višem menadžeru proizvoda za Workstation Division NCR-a, ugrađeni multiprocesuing kod *NT*-a lakši je za premeštanje i brži od onog kod *Unix*-a, zasnovanog na dodacima sistemu.

Ključ za podršku višeprocesorskom radu leđi *NT*-a je HAL (*hardware abstraction layer*) HAL deje ker nelu jednake kompletan uvid u hardver. bez obzira imao sistem jedan ili deset procesora. Razlikli ker diver ne zahteva različite kernele same različite verzije HAL-a. Sva *NT* aplikacija koje se izvršavaju na jednoprocesorskoj mašini radiće i na onoj sa više procesora. „HAL je drajver za kuliju". slikovito kaže Dave Becker. menadžer za marketing servera u firmi Wyse.



Višeprocesorski *NT* sistem ima arhitekturu se deljenom memorijom, u kojoj svaki procesor ima jednaku mogućnost pristupa zajedničkoj memoriji. Protokol se koherenciju *cache* memorije osiguvava da procesori koji pristupaju istim memorijskim lokacijama u svojoj *cache* memoriji imaju najnoviju verziju zajedničkih podataka.

NT ne radi na svim vrstama višeprocesorskog hardvera, on podržava samo simetrične višeprocesorske mašine. Sistem mora podržavati koherenciju *cache* memorije na hardverskom nivou i pojedinačni procesori moraju biti sposobni da izvršavaju isto kopiju niti ili sistemski servis nakon svakog interrupta ili bilo odziv na isto kojim situaciju. Ovaj drugi uslov ima jednu interesantnu posledicu. čak i ako vaše aplikacija nema više niti, još uvek se može brže izvršavati na višeprocesorskom hardveru. jer *NT* može paralelno kodu aplikacije izvršavali sistemske niti, kao što su rutine za servisiranje I/O zahteva.

Što sve obuhvata pisanje HAL-a? Prema Becker-u, potrebno je oko 30 000 linija C koda. Flanary iz NCR-a kaže da je pisanje HAL-a najveći deo posla oko prenosa *NT*-a na novu platformu.

Najveći deo višeprocesorskih sistema predviđe se kao skupi i moćni serveri. Sequent će na primer igrati vrlo važnu ulogu na tom tržištu. Neke kompanije vide i važnu ulogu višeprocesorskih mašina na radnom stolu. Posebno NCR koji planira svojim linijom računara zasnovanih na Pentiumu da napadne tržište procesorski intenzivnih aplikacija, kao što su CAD-ohrazla slika i slično izdvajajuće Wyse također vidi ulogu SMP na radnom stolu posebno u inženjerskim radnim stanicama. „Zadnji ekser u kovčegu u kojem se sahranjuje razlikovanje između PC računara i radnih stanica biće višeprocesorske PC se lokalnom magistralom koji radi pod *NT*-em" kaže Becker.

Proizvođači višeprocesorskih mašina veruju da mogu da se naći neće „vrućim" jednoprocesorskim radnim stanicama. Alpha serija, Mips R4x00MC i Pentium čipovi predviđeni su za višeprocesorske projekte i zbog toga „ave se svodi na odnos cena i performanse" kaže Flanary iz NCR-a. *NT* daje proizvođačima višeprocesorskih mašina platformu koja im dervelijavju da razvijaju neke koncepte radi pobeđivanja performanse a da sve vreme zadržavaju kompatibilnost sa širokom bazom postojećeg softvera.

Bob Ryan

INFORMATIČKI INŽENJERING I AUTOMATIZACIJA

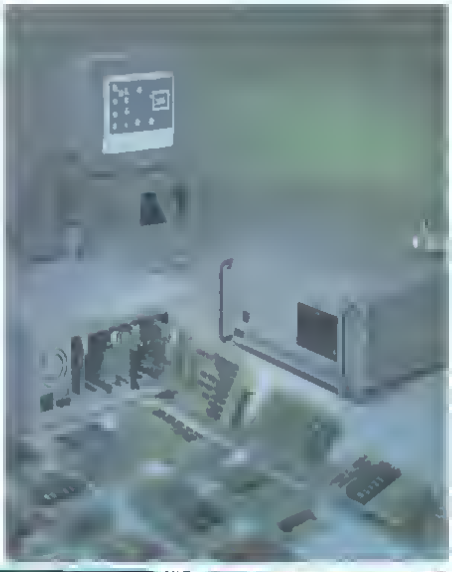
Računarske mreže pod
NOVELL NetWare
operativnim sistemom

Višekorisnički operativni sistemi
UNIX, XENIX, Open Desktop
SCO
THE SANTA CRUZ OPERATION

Mrežne kartice i komunikaciona oprema
LONGSHINE

NOVI SAD
VOJV. BRIGADA 7/IX
PO BOX 39
TEL-FAX 021 22 258
TEL 021 27 181

COMTEC



Kompletno rešenje za automatizaciju
industrijskih procesa i laboratorija
bazirano na PC računarima

PC-LabCard / PC-TooiCard
ADVANTECH
Industrial & Lab Automation with PCs

Software za akviziciju podataka,
praćenje i upravljanje procesima
LABTECH
Laboratory Technologies Corporation
Wilmington MA - USA

MIPS IZAZIVA INTEL NA NJEGOVOM SOPSTVENOM TERENU

„Mips Technologies" (Mountain View, California) relativno mala kuća u vlasništvu „Silicon Graphics" a koja se bavi projektovanjem integrisanih kola nada se da će *Windows NT* omogućiti da njihovi vrlo cenjeni RISC procesori dovedu u pitanje Intelovu deceniju dugu dominaciju na PC tržištu. Sirova brzina nije problem za Mipsx procesore. R4000 radi na mlarno advokatski neštanosot (50/100MHz) i košta otprilika isto kao Intelov 486 na 33MHz (oko $400 za količinu od 1000 komada) a otprilike ja dva do četiri puta brži. R4400 na 75/150MHz predviđen da uđe n masovnu proizvodnju u junn košta isto koliko i Pentium na 66MHz (oko $1000) a dvaput je brži od Intelovog najboljeg procesora.

Danas sn „Mips"-ovi procesori samo kap u bujici 80x86 kompatibilnih procesora koji se pušlejn na PC tržište svake godine. Intelova proizvodnja n velikim senjama celigurave zelav izvor čipova a njegovi konkurenti (AMD i Cyrix) pomažn da sa cene drža pod kontrolom. Mips je odgovorio time što je prodao i cen ce za proizvodnju R4000 procesora nezavisnim snabdavačima „Integrated Device Technology" „LSI Logic" „NEC Electronics" „Performance Semiconductors", „Siemens Componante" i „Toshibe"

Čak i eko Mips može da kontroliše ponudu i cene biće tešku uventi proizvođače PC klonova da je RISC PC isplativ. Projektovanje novih sistema oko brzih RISC čipova je težak zadatak. Mnogi proizvođači ia čunare establjani su prošlogodišnjim ralom cena i ne mogu sebi da dozvole velike izdatke za razvoj i istra živanja

Sejanje Mipsovog semena

Da bi ubrzao razvoj sistema zasnovanih na R4000 procesoru, Mips je nedavno osnovao kompe nliju ODC *(Open Design Center)* ODC prodaje komplete projekata koji proizvođačima klonova omogućavaju da načina skok od korišćenja 80x86 procesora do korišćenja R4000. Jeden od kompleta sadrži rafe rentnu osnovnu ploču za proizvodnju sistema posebno podešenih za Mips procesore drugi sadrži par ASIC *(Application Specific Intergrated Circuit)* kola, koje proizvođačima omogućavaju da R4000 prilagode postojećim 486 osnovnim pločama. I u jednom i u drugom slučaju treba samo dodati memorije periferije i *Windows NT* i dobili se kompletan R4000 sistem za koji Mips kaže da će nnistiti Pentinm

Jedan da prvih ODC ovih klijenata jeste „Carrera Computers" lak osnovana kompanja n Laguna Hillsn n Kalifornji. U februaru „Carrera" je počela de proda je osnovnu ploč sa R4000 procesorom zasnovanu na jednom od ODC-ovih referentnih projekata. Načinjena sa dimenzijama koja odgovaraju standardnom AT kućištu i namenjene močnim radnim stanicema ili servarima ova ploča vredna $4395 sadrži lokal-bas video kontrolar razclnice 1280x1 024 piksele 2MB video memorije podnožja za SIMM memorijske mednie kapacitete ukupno 256MB četin EISA ul ćnice SCSI 2 priključak Ethernet priključnak paralelne i serijske priključke i nlaze za miša i tastaturu po PS/2 standar du

Osnivač i predsednik „Carrera" Bruce Fausl veruje da će proizvođači PC računara prihvatiti RISC teh nologiju jer ja ekonomski čvrsta i tehnički bezbolna. Iako je „Carrera" osnovna ploča načinjena u skladu sa *ARC (Advanced RISC Computing)* specifikacijem Fensl kaže da na duže vreme „ARC neće vredeti ni pišljivog boba" Osob na ARC specifikacije kao što su EISA magistrala SCSI i ugrađn harilver za nmrežavanje mogu sa izostaviti iz maksimalno ogoljanih klonove koje njihovi proizvođači rado prodejn. U stvari „Carrera" npravo iedi na R4000 osnovnoj ploči niske cene. Fausl tvrdi da će krajem ove godine moći za $3500 de sa kupi RISC PC koji ima četiri puta bolje performanse od 486DX2 na 66MHz

Acer razvija PICA arhitekturu

Slično tvrdnju iznosi i Ronald Chwang generalni direktor „Acer America" (San Jose California) „Acer" medutim ne drugačij pristup. Umasto de pihvati projekat Mipsa kompanja je razvila novu sistemsku arhitekturu koju naziva PICA *(Performance-enhanced I/O and CPU Architecture)*. „Kako snaga procesora nastavlja de se povećava" kaže Chwang „mistela stalno pronalazn nova rešenje de bi propusnu moć I/O dela usklađli sa procesorom. *NT* nse ohrabruje da radimo npravo to"

U srcn projekta koji je nakk na mini računare je 64-bitne magistrala sa propusnom moći od 200MB/s, koja povezuje R4000 procesor memoriju video bed i 64 bitni četvorokanalnin I/O procesor koji obezbeduje osem nazivisnih DMA kanela *r bus mastering*. Inta grani *Ethernet* i SCSI I/O adapten vezani na ta kanale rade brže nego na EISA magistrali uz eliminisanje vi soka cena logika ze implamantacijn *bus-mastering*-a na EISA magistrali. Se druga strana standardni *AT bus* kontrolai podržava ISA kartice

PICA čip-sal i cenovne ploče mogn se naći na samo u „Acer" sistemima već će bti licencirani i drugim proizvođačima preko ODC Tipične konfiguracija sadržavala bi R4000 na 50/100MHz 16MB RAM 200MB SCSI tvrdi disk CD ROM nredaj floppy disk akcelatorsku video kartcu na posebnoj video magistrali ugrađani *Ethernet* i SCSI-2 port Cena takvog računara bila bi oko $3500 (bez monitora)

PICA će imati konkurencju u još jednoj novoj post-AT arhitektnri – Intelove PCI i *Penpheral Component Interconnect*. Ne samo nval VL-Bus standardu lokalna magistrale PCI je zapravo celoknjna sistem se arhitektira nova generacja koja baš slučajno podržava ISA i EISA magistrale zbog kompat-bilnosti nanže sa starim karticama za proširenje. Kao i „Acer" Intel nudi kompletan čip-sal koj rreba proizvođačima de olakša implamentanje nova arhitektura kao i „Mips" i ODC Intel nudi referentnim ploču koju svako može koprat ili modifikovati

Značajna razlka izmedu PCI i PICA ja n tome što Intelova PCI arhitektnra podržava bilo kej procesor ne samo Intelov ili 80x86. Sadašnje verzija Acerovog PICA projekta radi samo se R4000 senjom iako Acer kaža da će budnće verzje moći da podrže i druga čipove. Ono što je zajedničko za PCI i PICA jesu visoka performanse kompatibilnost nanže sa poznatom AT magistralom i balansirani pristnp arhitektun sistema

Windows NT odrednje kako će izgledati bojno polje ne samo za sledeću generaciju mikroprocesora već i za sledeću generacijn arhitektnre PC sistema.

Tom R. Halfhill

Možda još može biti važne uloge za olevku ili sistem sa grefičkom ieblom za bolne programa za obradu teksta grafike- tabelarne proračuna i baze podstaka koji se razlikuju od PDA sistema *(Personal Digital Assistant)* koj se koriste kao sgande i za komuniciranje. Razmišljajte o sistemu sa olevkom kao o bloku i o PDA kao džepnom računaru. Imala i jedan i drugi i konstite ih u različite vrama za različite namena. Ako se sistemi sa olevkom prošire *Windows for Pens* će profitirati zbog svojih porodičnih veza. On dodaja novi kvalitet standardnom *Windows* softveru koji se izvršava na sistamn sa olevkom i nudi pisclma softvara već poznato programersko okruženje. Međutim na tržištu kojim dominira upotreba za prikupljanje podataka na terenu ovakvi oblici saradnje nlan uvek potrebni

BRŽIMO MALI

Saraunju ja još teže otkriti u slučaju najmlađeg člana *Windows* porodice *Modular Windows* izvlšavajući se n megabaju ROM e u Tandy Video Information Sy stam-u (CD ROM plejer sa 256 procesorom) *Modular Windows* prikazuje slku na TV aparatu i komande pn ma iz daljinskog npravljača sa osam tastera izbačeni su delovi *Windows*-a nključujući *Program Manager* mnltitasking prozore koji se preklapaju menje u nemen da se postignn dve stvari da se *Windows* ugura n jedan megabajt i da se grafičk interfejs pojednostevi za TV konsnika. Ono što ostaja jesta oko dva tređine *Windows* API i deo kontrole konsničkog interfejsa (kan katurne verzije tastere i polja za skrolovanje) I API ja (za kontrolu daljinskog upravljača direktni video prietup i Tandy VIS memorijske karte)

Program za *Modular Windows* sistem ima porodičnih sličnosti sa običnim *Windows* programom ali samo u drugom kolenn. *Modular Windows Software Development Kit* nporučuje se sa posebnom verzijom igra Reversi (ruski šah) koje se može kompajlirab ze obe platforme ali samo zahvaljujući velikim delovima uslovnog koda koji su specifični za *Modular Windows*. Adaptirani *Visual Basic* koji ja uključen n SDK može de ubiazi raz lika I najnovji paket alata za razvoj multimedia aplike cija poslužice za razvoj programa za *Modular Windows Windows 3 1/MPC* i *Sony MMCD* plajer na osnovu istog izvornog koda. I pored ove pomoći pisci softvera za kućnu zabavn će možda na kraju uslanovit da srod nost *Modular a* sa običnim *Windowe*-om nije toliko karl sna kao što tvrdi Microsoft samo zbog toge što su TV i računan još uvek sasvim različite zverka

Vrednost ovakvih porodičnih odnosa poslaje sve manja ako kao Microsoft zamisite de će sa *Modular Windows* konstil za kontrolu inteligentnih prijemnika za kablovsku televiziju moćne *fax* maline i fotokopire i sistema za automatizaciju domaćinstva. Microsoft ne želi da kaže ko u ovom trenutku eksperimeniše sa *Modular Windows*-om u ova avrhe e nijedan od kandidala takode ništa na priča.

Iako će ovakvi uređaji izvršavati programe koji ni izdaleka ne liči na standardna *Windows* aplikacije Craig Mundie generalni direktor Microsofte za napisdna kontenička tehnologije tvrdi da je upravo sroditstvo sa *Windows*-om kritično. On nabraja pozniste progra marske alata procedura kao faktore koji će ubrzali vre ma za koje će *Modular Windows* softver slizeti na tržiš te. To može biti tačno- ali još nvak *Windows* nije ni na koji način pokazao da može uspešno opsluživati to tržiš te. „Dovoljno nam je lasko da nbedimo korisnike da

Kome treba *Windows NT*?

Zahtavni korisnici

kojima je potrebna atablilna platforma na kojoj će se u multitaskingu izvršavati 32 bitna *Windows* aplikacija kao i DOS i 16-bitne *Windows* aplikacije

Administratori lokalnih računarskih mreža

kojima je potrebna robusna, proširiva platforme za konščenje sa 32-bitnim serverskim aplikacijama

kariste DOS", kaže Vincent D Amico direktor razvoja softvera za „MicroTouch" (Wilmington Massachusetsj proizvođača ekrana osetljivih na dodir koji je skoncentrisan na aplikacije za igra i elektronske kase (*posti-of-sale*)

Iako usamljeni primer Tandy VIS nređaja i Modular *Windows*-a na njemn na govori mnogo u prilogveliloh tvrdnji o mogućnosti ze skalabilnost *Windows*-a nanže Microsoft i dalje preuzima sariju inicijativa za smanjiva nje i prilagođavanje *Windows*-a za nomadskn upotrebu Vramena name ze gnbjanja. Nje slučajno da je *Mac PowerBook* postao mobilna mašina broj jedan za poslovneg čovaka na puln. Sa privlačnom kombinacijom softvera rezidentog u ROM-u elegantno načinjenog hardvera i sposobnosti za jednostavno končenje faksa i umrežavanja. *PowerBook* postavlja definiciju onoga što *Windows* mašine treba tek da postignn

Microsoft u razvoju navodno ma još jednu verziju *Windows*-a za prenosne sisteme *Mobile Windows*, ik kratko *MobiWin* namenjen je *notebook* i *subnotebook* računarima. Imaće dodatne mogućnosti kao što je *hot docking* pri postavljanju n kući bazni sistem kojim se automatski sinhroniznin konfiguracije prenosnog dela i beze kso i osobina da su idaloteka n skladu sa standardnim *Windows*-om kaže Gerry Purdy potpredsednik i glavni analitčar za eblast mobilnog računarstve za „Dataquest" (San Jose California)

WinPad je grafički interfeja za ručne PC računara ili *hand-held* PC ja. On koristi one što Purdy opisuje kao „litlove i spratove" Na primer računar na radnom stoln poseduje rasurse za upravljenja prozornima koji se preklapaju. Kade skran postane mali više nemata taj luksuz. Umesto toga, slažete prozore jedan na drugi i brata ih pritiskom na tastere „lifta"

PUKOTINE U KONTINUUMU

Ono što slogan „*Windows* svuda" sada uglavnom znači jeste da *Windows* izvršava i aplikacije koje pripadaju serverima kao i one ze radne stanice Microsoft

Postoje li razlozi zbog kojih niste kupovali sako u delovima?

Sve možete kupovati u delovima.
Teoretski, da.

Sve se može lepo sastaviti.
Teoretski, de.

I evi će delovi, kada aa sastave, činiti jednu savršenu celinu koja će Vam potpuno odgovarati.
Prektično, ne.

Iz istih razloga nemojte kupovati ni računare u delovima.

MI na prodajemo delove.
MI prodajemo RAČUNARE.

COMTRAD računari su dugo tražena kombinacije savršenog sastavljanje delova u kompaktna cailne.
Keda raeklopite COMTRAD, dobićete kvalitetne delova, keda negde kupujete računer u delovima nećeta dobiti celinu kvalitetnu kao COMTRAD.
Čudno je, ali je tako.

I teoreteki I praktično.

Kupujte računare a ne delove.

Combinacija
Tražena
dugo

ComTrad

Genex epartmani
Vladimira Popovića 6

011 - 222 - 41 - 51
011 - 222 - 26 - 52
011 - 222 - 41 - 39

takođe želi da se pod *Windows*-om radi i umrežavanje, ali to je lakše reći nego učiniti

Unatoč zgodnih mrežnih sposobnosti koje nude *Windows for Workgroups* i *Windows NT*, ne možete uspostaviti veliki mrežni sistem na nivou kompanije ako kupujete samo kod „Microsoft"-a. Šta nedostaje? Najviše se primećuje nedostatak distribuiranog globalnog servisa direktorijuma za praćenje svih ljudskih hardverskih i softverskih resursa organizacije. Takođe, *Windows* ne daje servise direktorijuma. Čak ni *NT* server na kojem će postojati administrativni domeni i „domeni povezivača" u *LAN Manager* stilu neće moći da se meri sa sposobnostima Novell-ovih NetWare Directory Services, ili servise direktorijuma u ONC (Open Network Computing) i DCE (Distributed Computing Environment) stilu koji su rasprostranjeni u *Unix* okolinama.

„*NT* će imati servis za dodelu imena baziran na domenima" kaže Jamie Lewis, analitičar u firmi „Burton Group" (Salt Lake City, Utah) „ali ne i pravi globalni katalog". Podrška za DCE RPC (*remote procedure calls* ili pozivi procedura se drugih mašine) u *NT*-u omogućava pisanje softvera da raspored servise između *Unix* i *NT* platformi – ali kako će se obznanjivali i locirali ovakvi servisi? *NT* u početku neće podržavati DCE servise kataloga. *Cairo* buduća verzija *Windows*-a u kojoj će u distribuirani objektni sistem biti prepleteni OLE, RPC i sistem datoteka *NT*-a, takođe neće tako skoro rešiti problem kataloga

Dodajte ovom tehničkom izazovu i Microsoftovu istorijski potvrđenu nesposobnost da mrežni softver prodaje korporacijama koja se još jednom potvrđuje slabom prodajom *Windows-a for Workgroups*. WFW je zapravo dragulj čija se vrednost ne ceni. Ljudi koji se bave MIS uslugama (*Management Information Services* ili obrada podataka bitnih za upravljanje firmom) imaju loše mišljenje o njemu jer se boje da će *peer* mreža ugroziti centralizovano upravljanje podacima, ali su oni propustili da vide u čemu je poanta. Server je tu opciono pogodnost

Prava vrednost *WFW* leži u programima koji su namenjeni radu u grupi, kao što su *Mail*, *Schedule+* i *MAPI* (*Messaging API*, funkcije za prenos poruka) koji dodaju nove (i to centralne kontrolisane) funkcije poslovnoj lokalnoj mreži zasnovanoj na serveru kao što je *NetWare*. Zašto je ova poruka ostala nezapažena? Zato što je niko nije preneo. Microsoft u nedostaju „Novell"-ova vojska prepredavaoca i bataljon direktnih prodavaca kakav poseduje „Sun Microsystems"

U slučaju *NT*-a, vest je još bolja. Osnovni proizvod, na primer, obuhvata usluge daljinskog pristupa, koje mogu da barataju asinhronim i čak X.25 vezama. Podrška za X.25 protokol je naročito važna bilo kojoj kompaniji koja želi da mrežne servise proširi na svetskom nivou koristeći mrežu sa komutacijom paketa. „Bez obzira koliko je umrežavanje pod *NT*-om dobro" kaže Lewis „Microsoft neće zagristi u Novellov kolač sve dok ne formira svoje sopstvene kanale za prodaju"

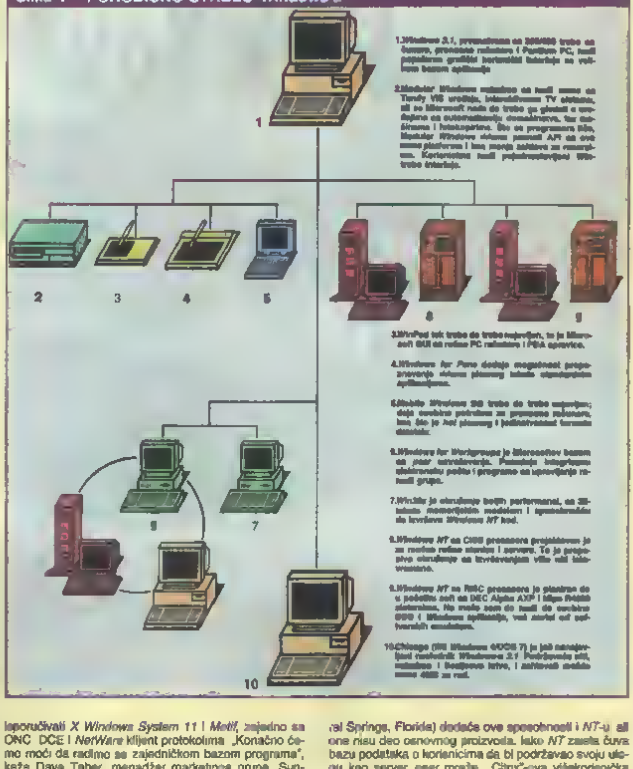
UJEDINJENI *Unix*

Vodeći prodavci *Unix* sistema uključujući HP, IBM „The Santa Cruz Operation", „Sun" i „Univel" nedavno su se pobratimili i proglasili kraj ratovima oko umrežavanja i sistema za prozore koji su razjedinjavali *Unix* zajednicu. Svi ovi proizvođači sada obećavaju da će isporučivali *X Windows System 11* i *Motif*, zajedno sa ONC DCE i *NetWare* klijent protokolima. „Konačno ćemo moći da radimo sa zajedničkom bazom programa", kaže Dave Taber, menadžer marketinga grupe „SunSoft" (Mountain View, California). Može izgledati ironično da je jedino pojave *NT*-a mogla prethodni ovom davno zakasnelom potezu, ali ostaje činjenica da će ujedinjeni *Unix* – ako se materijalizuje – biti sila na koju će morati da se računa.

NT-u nedostaje višekorisnička osnova *Unix*-a, kao i njegova sposobnost da se preko asinhronih i mrežnih priključaka prenose i tekstualni i grafički orijentisani korisnički interfejsi. Nezavisna firma, „Citrix Systems" (Cereal Springs, Florida) dodaće ove sposobnosti i *NT*-u, ali one nisu deo osnovnog proizvoda. Iako *NT* zasela čuva bazu podataka o korisnicima da bi podržavao svoju ulogu kao server *peer* mreže, „Citrix"-ova višekorisnička verzija *NT*-a je veliki zahvat koji unapređuje upravljanje sesijama i pravima korisnika kod *NT*-a, optimizira sistem za dodelu vremena procesima i implementira preteksle za tekstualne i grafičke terminale

Ako Microsoft može s pravom tvrditi da *Unix* nije iz temelja građen za simetrični multitasking proizvođači *Unix*-a mogu sa jednakim pravom istaći da *NT* nije projektovan da podržave više korisnika prijavljenih se udaljenih terminala. Ako mislite da u devedesetim te neće be-

Slika 1 – PORODIČNO STABLO *Windows*-a



Slabosti *Windows*-a

Koliko je mašina spremno za *NT*-a

Datasquest procenjuje da je blizu 11 miliona računara klase 486 prodato tokom 1991 i 1992. Ove predstavlja pukih 7 7 procenata instalirane baze PC računara koji bi mogli biti spremni za *NT*. Neodređeni broj tih 486 računara, međutim, ne može izvršavati *Windows NT* bez dogradnje memorije ili povećanja kapaciteta masovnih memorija

Unlock Your Mind

LICENCNI SOFTVERI

RAČUNARSKI SISTEMI

KOMPONENTE

ŠTAMPAČI

Budimska 10a, 11000 Beograd; Tel/Fax: (011) 337-120, 340-227; pejdžer: 2222-302, #987

no, pokušaje da svom lokalnom prodavcu video-opreme *prodate upgrade na Xenix* ili *Windows LAN*.

Unix se takođe čvrsto drži u umrežavanju i distribuiranoj obradi. *NT* svoj dobar prijem kod poznavalaca mreža uvelike duguje usvajanju *Unix* mrežnih protokola kao što su TCP/IP utičnice (sockets) i DCE RPC. Međutim umrežavanje pod *Windows*-om još uvek nagnije *NetBEUI* protokolu koji ne poznaje rutiranje i mora preći dugačak put pre nego što će moći da se poredi sa kompleksnošću i sveskim domstima *Unix* umrežavanja.

FAKTOR IBM-a

Kao i Microsoft IBM će obeleđanili porodicu prenosivih i nepreniosivih operativnih sistema „Sa tako velikim izborom hardvera" kaže John Soyring direktor programa za razvoj softvera „nijedan operativni sistem na može sva pokriti" DOS i OS/2 su maksimalno podešeni za Intel procesore kaže on dok se AIX dobro sla že sa RS/6000 IBM takođe vidi potrebu za prenosivim sistemom zasnovanim na mikrokernelu koji će moći da se testira u beta lazi kasnije ove godina.

Najavljeni *Workplace OS*, zasnovan na derivatu *Mach 3.0* sistema veoma podseća na *NT*. Ujedinićе više ikova – u najmanju ruku DOS 16-bitni *Windows*, OS/2 i AIX – pod budućom verzijom sadašnjeg *Workplace Shell*. *System Object Model* nove objektna tehnologija koja pogoni proširivi *Workplace Shell* podržavaće distribuirana objekta koji komunicirаju koristeći TCP/IP utičnice ili DCE RPC. Za to vrama IBM nastavlja da poboljšava OS/2. Verzija 2.1 koja će se isporučivati pre *NT*-a dobija 32-bitnu grafiku veoma potrebne video i CD-ROM drajvere i munjevito brzu podršku za *Windows 3.1*.

Dilema koja se postavlja pred IBM-om jeste da su još uvek retke ubičajena aplikacija za *Presentation Manager* koje koriste napredne osobine sistema. Kao razultat korisnici smatraju OS/2 za platformu na kojoj mogu izvršavati DOS i *Windows* softver. Na nesreću *Windows* je meta koje se brzo kreće. Zbog toga što OS/2 2.1 na podržava VxDs (drajvera za virtuelne uređaje) *Windows*-a *3.1* neće moći da izvršava *Win32s* aplikacija. Teoretski bi *Workplace OS* mogao da dođe i *Win32* lik, omogućavajući da se izvršavaju programi kao što to čini i *NT*. Soyrin tlaje nagoveštaje de to može biti plan za slučaj opasnosti ali dodaje da „oni /IBM/ neće ohrabrivati razvoj *Win32* softvera"

Iako ja položaj IBM-e na tržištu stonih računara uz

GRAFIČKI DEO ZAOSTAJE, ALI NE JOŠ DUGO

I pored *Windows*-ovoj grafičkoj ilka njegovom GDI (Graphical Device Interface) zasnovanom na rastarskoj grafici nedostaju vektorske i poligonske primitive i 2-D i 3-D programske apstrakcije neophodna za zahtevnji ilustratorski rad CAD/CAM i aplikacije za grafičko predstavljanje. Takve primitive i apstrakcije služa za dve stvari pojednostavljuju programiranje i daju mesta mnogo boljoj hardverskoj podršci nego što je daju današnje lokalna magistrale i akceleratorske video kartice.

Poboljšanja GDI u *NT* u – putanje Bezijeove krive dvodimenzionalne transformacije – pomažu ali malo *Windows* može ući u arenu 3-D grafičkih radnih stanica na ramenima *NT*-a ali da bi se tu uspešno takmičio mora razviti bolji grafički API.

Dobra vest je da je Microsoft kupio licencu za *OpenGL* od Silicon Graphics. Loša vest je da *OpenGL* neće biti uključen u prvu verziju *NT*-a. Microsoft i Silicon Graphics rade na tome da se *OpenGL* i *Windows* GDI integrišu, ali to je vrlo složan zadatak koji će se verovatno raditi po lazama. Neki od proizvođača ne mogu da čekaju. „Mi ćemo *OpenGL* imati u našoj prvoj verziji" kaže Tom Steele iz firme Intrasgraph. NEC Technologies (Boxborough Massachussets) će takođe ponuditi podršku za *OpenGL* za svoju novu radnu stanicu zasnovanu na R4000 procesoru i *Windows NT*.

Mike Mitsch menadžer grupe za marketing računarskih sistema u NEC u vidi potrebu za komercijalnim softverom baziranom na *OpenGL* interfejsu čak i na hardveru koji nema grafički pipeline. „Berzanski mešetari žele da vizualizuju ponašanja stotina paketa deonica po više koordinata, a GDI uopšte nije rađen za takvu primenu", kaže on.

Win32 aplikacija koje budu pisane za *OpenGL* moći će ovakve zadatke da izvršavaju sa relativnom

***Windows* GDI ne odgovara se aplikacije kao što je ova simulacija protoka vazduha oko efera prikazana ovde dok radi na RISC platformi (fotografija ljubaznošću Research Systems Boulder, Colorado)**

lakoćom i na platforma koje imaju akceleratora za *OpenGL* moći će to rade u realnom vremenu. *OpenGL* akceleratori na nivou kartice za RISC računare su vrlo verovatan pravac razvoja. Ako se GDI i *OpenGL* stvarno uspešno ujedine – e Bill Glazier iz Mipsa naglašava da „nece prihvatiti nikakve budževne" – *NT* može zaista postati „srleško operativni sistem za Silicon Graphics radne stanice.

Oslaje činjenica da prva verzija *NT*-a neće imati grafiku visokih performansi. Niko se međutim ne žali. Proizvođači sistema i pisci softvere misle da će odnos između Microsofta i Silcon Graphicsa odvesti grafički deo *NT*-a u pravom smeru.

drman, kako kaže Anders Vinberd viši potpredsednik razvoja za „Computer Associates" (Islandia New York) Microsoftova snaga na polju korporacijskiri informacionih servisa je još manja „Ljudi u PC industriji imaju grazrive u svom pomanju sveta kade se govori o CICS-u /*Customer Information Control System*/ i obradi transakcija" kaže on.

Međutim uloga OS/2 kao sredstva za povezivanje PC i *mainframe* računara ostaje ključna prednost. U ovom carstvu IBM je taj koji izgleda kao inovator. Microsoft preuzima konvencionalni prestup poklanjajući DLC (*Data Link Control*) protokol i jednostavni 3270 emulator uz svoj *NT* te odvojeno nudi SNA (*System Network Architecture*) i *NetView* servise IBM koji je već uveliko u toku sa svim stvarima povsuje ulog pri pramajući paket DCE proizvode ze OS/2. Kompletniji od onog što *NT* nudi paker će omogućavati distribuiranu obradu transakcija i kantrolu OS/2 AIX i resursa *mainframe* računara.

ARHITEKTE *NT*-a GOVORE

Dave Thompson i Lou Perazzolli su ključni arhitekti *Windows NT*-a, oni su menadžer razvoja za umrežavanja odnosno baznog sistema. Obojica su radili na DEC VMS sistemu pre nego što su došli u Microsoft. Perazzolli je zepravo bio i deo tima za razvoj VMS-a, na Elu se Davidom Cutlerom koji je sada glavni arhitekta u Microsoftu. Šel BYTE-ovog urada na zapadnoj obali Intervjuisao ih je za ovaj članek

Interesuje me pomisao da ste imali mogućnost da pogledate unatrag na istoriju srednjih i velikih sistema i izaberete osobine operativnih sistema koje treba pranati na sistema nižeg ranga

Perazzolli: To je ono što smo uradili. Gledali smo *Unix* standarde i nake od stvari koje su ljudi radili se bazama podataka na mainframe računerima. Mnogo smo toga radili u DEC-u jer amo tamo razvijali novi operativni sistem i znali smo koje ograničenja imaju se VMS-om. I koja ograničenje su imali as njihovim *Unix*-om. Mnogo smo saznali o toma šta ljudi traže.

Imali smo interesantnu priliku da stvari razvijamo iz tamalja, da sve pišemo od nula. Uzimali bi kod iz takućih Microsoftovih projekate kada smo smatrali da je to dobra ideja ili bi prepisali stvari kada smo mislili da će to biti dobro

Da li ste imali utisak da radite na istorijskom operativnom sistemu kaoa ste počinjali rad na ovom projektu?

Perazzolli: Jedna od stvari koja amo htel da uradimo bile ja da na PC donesemo operativni sistem visog ranga. Mislili smo da će se to zavliti u zaista veliku stvar jer smo videli potrażnju za nečim takvim. Videl smo da se dešave trend smanjivanja. I videli smo da će mnini raunar biti zamenjen PC jem. Činila nam se da ja ono što radimo logična stvar. Plus, kli jent/server je tek počinjao da biva interesantan mnego osnovnih stvari je već bilo urađeno. I ljudi su već počeli da misle o klijent/server arhitekturi kao modelu za vođenje njihovih poslove. Dakle ono što smo želeli da radimo bilo je da projektujemo nešto što je robusno. I imalo osobine koje su ljudi želeli: I klijenti/ servai sposobnosti

Thompson Moj pogled je možda nešto drugaciji jai sam ja došao spolja. Moj *background* je bio više VMS-baziran nego *Unix* III čak PC. I bilo mi je jasno de je za PC potreban robustan operativni sistem vi sokog ranga. Bilo mi je jasno de ako iko to treba da to napravi de ja Microsoft u najboljem položaju. Tako da sam svakako osećao potencijal od samog početka

Da li je ovaj današnji NT relativno spreman za Cairo? Da li je projektovan na takav način da je tak prelazak na model koji je više objektno orijentisan?

Perazzolli: Da. U stvari jedne grupa ovde u Microsoftu radi na razvoju. I u osnovi sve što oni rade je da dodaju objektno orijentisan sistem datoteka *NT* već ima četiri sistema datoteke /FAT NTFS HPFS CD ROM/ pa dodavanje novog sloja nije teške stvar. Većina ostalih stvari u *Cairu* su na korisničkoj strani – na primer, poboljšanja koja OLE 2.0 donosi u izvršavanju aplikacija

Šta je vaš sledeći zadatak?

Thompson: U služaju umrežavanja postoji beskrajna serija zahteva za razne mogućnosti povezivanja sistema. Postoji izvestan broj načina na koji se sistemi povezuju. I jesno da ja jedna od stvari koja ćemo raditi proširanje podrške na tom polju. Prame toma mislim da je naš rad na umrežavanju već unapred zacrtan"

GOSPODO, POLOŽITE SVOJE OPKLADE!

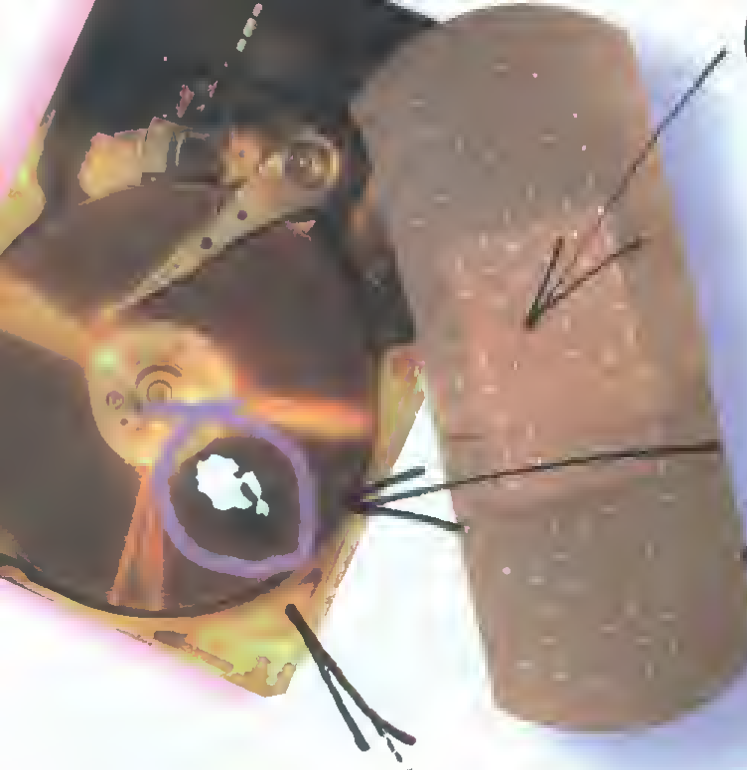
Da bi se dobio objektivan pogled na tekmičenje operativnih sistema, dobro je porazgovarati sa proizvođačem softverskih alata koji podržava sve sisteme *Universal Componeni System* firme „Software Transformations" (Cupertino California) omogućave pisoma solftvera da pišu aplikacija za *Windows* Macintosh OS/2 i više vrsta *Unix*-a. „Sto posto naših konsnike zainteresovano je za *Windows*" kaže potpredsednik STI zaduzen za projektovanje Doug Donzelk „Izbor sa drugu platformu ja *Unix* Mac i na kraju OS/2"

Gotovo da nika ne može zanemarti snagu i privlačnost porodice *Windows* operativnih sistema. Kao rezultat, mnogi strahuju da će Microsoft poštiti istu vrstu kontrole kakvu je nekade imao IBM. Bez obzira što će se to svakako dogoditi tvrde drugi to neće biti tako loše za korisnike. Vladavina IBM-a nam je dala pouzdanu infrastrukturu za banke sistema za rezervaciju mesta u avionima i druge elemente modernog života. Možda nam je lekav standard ponovo potreban i možda je taj standard *Windows*

Ali na sreću ili na nesreću stvari neće biti tako jednostavne. Uz uspon *Windows*-a svi osecaju takmičarski žar. Pranosivi sistemi zasnovani na mikrokernelu su glavna stvar i dok je *Windows NT* na čelu trka nije tako daleko od ostalih kao što Microsaft želi de nam prikaža. *NetWare* OS/2 *Solans NextStep* i *Mac* takođe mignuju na nove platforme i utiču na *Windows* isto onoliko koliko *Windows* utiče na njih. Igra zvana „*Windows* svuda" donosi veliku zaradu proizvođačima i kancinicima, al to neće biti jedina igra.

John Udell Byte June 1993

Preveo *Zoran Kehler*

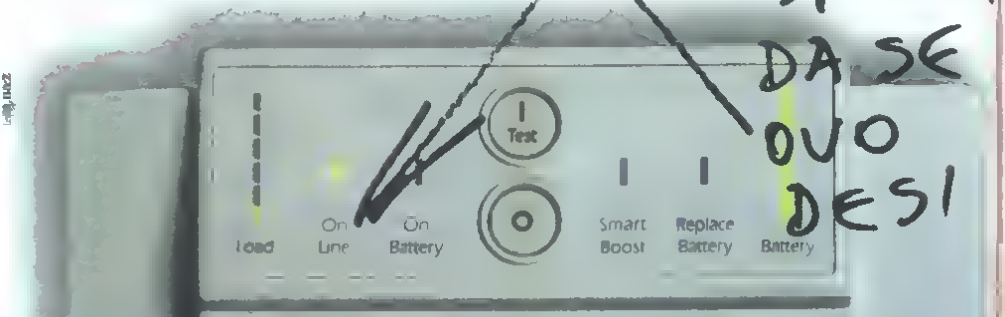


OVAJ FLASTER NEĆE SPASITI OVAJ DISK...

ups

OVAJ U.P.S ĆE SPREČIT DA SE OVO DESI

Sigurna preventiva protiv hirova elektrodistribucije



SAGA 11070 BEOGRAD M[illegible]OVIĆA 9 SAVA CENTAR, TEL 011/222 3579 147 182, 222 4323 EXT 256 & 259, 222 4322 EXT 256 & 259 FAX 011/455 785, 605 578

ZVEZDA U OFISU

Matrični štampači su odavno dostigli svoj tehnološki zenit, i iz te oblasti već godinama ne dolaze gotovo nikakva uzbuđenja. Daleko od radoznalih očiju kompjuterska štampe, oni stolički obavljaju svoj ropski posao po kućama i kancelarijama, atrplijivo izbacujući na stotine stranica dnsavno razne dokumentacije. Zašto se ponovo vraćamo ovoj reklo bi se potpuno potrošenoj temi?

Jovan Regusek

Svojim ogromnim dimenzijama (taman toliklm da može da primi papir A3 formata) i malom razoluclJom (slova se devet igllca) Star ZA-250 je, na prvi pogled, gotove tipičan predstavnik glomazne klase kancelarijskih štampača. On, međutim, ima i nekoliko izuzetnih osobina (deklarisana brzina od 420 znakova u sekundi je prva na ovoj listi) koje ga toplo i ozbiljno preporučuju svakome ko razmišlja da automatizuje svoje kancelarijsko poslovanje ili obradu tehničke dokumentacije.

FONTOVI I SLOVA

Štampač Star ZA-250 je kompatibilan se Epson i IBM štampačima i time je praktično sve rečeno – teško je naći DOS i *Windows* softver sa kojim ova mašina ne bi mogla da sarađuje. Bilo da su u pitanju obrada teksta, rađne tabele baze podataka, poslovna grafika ili knjigovodstveni i administrativni programi svaka vrsta

U štampač je ugrađeno šest rezidentnih slllova slova (*Draft, High-Speed Draft, Sanserif, Courier, Script, Orator*) pri čemu sa neproporcionalna plema mogu dobiti u četiri različite širine – *Pica* (10 CPI – characters per inch znakova po inču), *Elite* (12 CPI) *Condensed pica* (17 1 CPI) i (20 CPI) *Pica* pismom može se odštampati 136 znakova a *Condensed elite* 272 znaka u jednom redu na strani A3 formata – korisna pogodnost za one koji rade sa velikim tabelama. *Elite* je nešto skromniji oblik od *Pica* – matricu slova Star ZA-250 formira za 8×11 tačaka u draft i 10×23 tačke u NLQ (*Pica*) odnosno 6×8 i 12×10 tačaka (*Elite*) – i otuda štampač sa tim tipom slova razvija nešto veće brzine

Svaki font sa javlja u više setova – tradicionalna *International* se 16 i savremenija *IBM Code Page* grupa sa 8 skupova fonteva, uključujući i kodne strane 437 (USA) i 850 (*Multilingual*) – koji pokrivaju primenu štampača u različitim govornim područjima, ali na i u našem jeziku. Srećom uvek ostaje mogućnost softverskog definisanja fontova i njihovog spuštanje u štampač. U Star Za-250 se može spustiti 255 slova a metod njihovog kreiranja je u priručniku sasvim dobro objašnjen

REŽIMI ŠTAMPE

Star ZA-250 obezbeđuje jednosmernu ili dvosmernu štampu sa logičkim traženjem (*logic-seeking*) i može sa podesiti za rad u jednom od tri osnovna režima – *High-Speed Draft, Draft* i *Near Letter Quality* (NLQ – veoma fini ispis, "skoro kao na pisaćoj mašini") Pri tome je svaki režim primaren određenoj vrsti posla – jedni se koriste kada stranicu treba odštampati što brže (fakture, poslovni izveštaji i uopšte poslovna dokumentacija), a drugi onda kada ima treba da izgleda što lepše (poslovna prepiska)

U bzpom režimu Star ZA-250 razvija fantastičnih 420 za *Elite* odnosno 372 znakova u sekundi (cps – characters per second) za *Pica* fontove. Ovu izuzetno visoku brzinu koja je približava izv *heavy duty* klasi (profesionalna klasa za najteže administrativne poslove) štampač postiže pojednostavljenim štampanjem slova uz neminovno ograničavanje kvaliteta otiska U normalnom režimu, uz kvalitet otiska koji je standardan za devetoiglične matrične štampače Star ZA 250 postiže 336 (*Elite*) odnosno 280 cps (*Pica*) a u NLQ režimu 84 (*Elite*) odnosno 70 cps (*Pica*) Normalne brzine se kreću u okvirima klase kojoj ovaj štampač pripada dok je u vežnji tečom brzinom Star ZA 250 letinski šampion

Za štampanje grafike obezbeđeno je osam različitih režima Najbrža štampa se postiže u *normal* režimu ali i uz najnižomatnji kvalitet otiska od samo 60 tačaka po inču Za najbolji otisak je potrebno izabrati *Quadruple* režim koji obezbeđuje 240 tačaka po inču, ili *Double-density plotter graphics* se 144 tačke po inču

KONTROLNI PANEL

Upravljanje uobičajenim funkcijama u toku štampe – pauza u radu, način dotura papira (brzo ili sporo napred ili nazad) postavljanje glave na vrh strane izbor pisma i širine fonta, izbor režima reda – i podešavanje štampača se vrši tasterima na kontrolnom panelu koji mogu imati više funkcija Podešavanje se mora obaviti za svaki od dva moguća režima (standardni i IBM) – parametri koji se odaberu u jednom režimu na utiču na rad štampača u drugom – ali je to dovoljno učiniti samo jednom Štampač je ako mu se to izričito naloži u stanju da trajno upamti sve odabrane parametre I na samo to izabrani parametri se mogu zaključati da bi se sprečile njihova promena iz neka aplikacije softverskim putem

Uz elementarne opcije preko kontrolnog panela se može pristupiti i mnogim drugim na prvi pogled nevidljivim parametrima kojima se određuje rad i način ponašanja štampača – vrsta emulacije način korišćenje RAM-a od 32 Kb tehnika štampe (jednosmerno ili dvosmerno kretanje glave) izbor kodne tabele i kodne strane, kvalitet štampe način dostavljanja papira, format i vrsta papira, broj redova na strani Star ovu vrstu podešavanja naziva *memory switches*. Ulaskom u ovaj režim podešavanja prelazi se kroz čitavo stablo komandi i funkcija, a kompletna procedura se obavlja odgovaranjem na pitanje koja štampač ispisuje na – papiru izabrane parametre štampač pamti do sledeće promene.

PARKIRAJ SA STRANE

Star ZA-250 prima jednostruki i višestruki (original + dva ili četiri kopije) preseviјeni perforirani (*fanfold*) papir pojedinačne listove širine između 101 i 406 mm odnosno sečene pojedinačne listove širine između 101 i 418 mm što uključuje i standardne A4 ili A3 formate kao i formate za samolepljive nalepnice Uz štampač se opciono isporučuju i jedinstveni ili dvostruki magazin za sečeni papir (*Automatic Sheet Feeder*), koji prima do 150 listova.

Rukovanje papirom je uskađeno sa tipičnim kancelarijskim potrebama Većina poslova se štampa na perforiranom papiru a pojedinačni listovi se koriste samo s vremena na vreme U ovakvim situacijama iz štampača Star ZA-250 nije potrebno izvaditi perforiran paper da bi se ubace pojedinačni list Ako se perforiran papir "udinra" u tzv *push feed* režimu štampač će ga prilkom uvlačenja pojedinačnog lista papira automatski parkirati van domašaja glave da bi ga, uz nekoliko trivijalnih operacija i pritiskom na (jedno) dugme ponovo postavio u položaj za štampanje

Štampač koristi standardnu Star crnu ili traku u boji koja se pokreće zajedno sa glavom Radni vek crne trake je 4 miliona znakova (*draft pica*) odnosno 2 000 stranica teksta A4 formata, a traka u boji čeln puta kraći. (Štampa u boji vas neće fascinirati – ona je taman dovoljna da se stekne brojka ispod oka!) Glava se devet igllca može da izdrži 100 miliona draft slova, odnosno 50 000 stranica. Da bi se očuvala preciznost štampe omogućeno je njeno elektronsko podešavanje sa kontrolnog panela na polašpanje dva susedna prolaza, a na kraju radnog veka – veoma jednostavna zamena.

JEDNO UKSUSTVO

Od dobrog kancelarijskog štampača se očekuju tri stvari da bude dovoljno brz da se na njemu u toku dana blagovremeno bez zastoja i odlaganja obave svi neophodni administrativni poslovi da bude dovoljno robustan da izdrži neprekidan rad i rukovanje više različitih ljudi i, konačno da bude dovoljno pouzdan i trajan da mu se mogu poveriti oni poslovi od kojih direktno zavise vaši prihodi

Deklarisana brzina od maksimalnih 420 znakova u sekundi je pri samom vrhu u svekolikoj familiji matričnih štampača i njegovi korisnici neće imati razloga da se žale na sporost. Tom brzinom Star ZA-250 će napuštajući matrični cenovni razred (cena reda veličine 1500 DEM) polako zalazi u područje znatno skupljih profesionalnih *heavy duty* štampača (brzina reda veličine 500 cps, cena 3000-5000 DEM) Niemo na žalost, bili u prilici da simultanim stroge kancelarijske uslove za koje je štampač namenjen (nezavisno smo nešto prema svakom hardveru) pa stoga i ne možemo da sudimo o njegovoj pouzdanosti i održljivosti u višemesečnoj intenzivnoj upotrebi ali imamo jedan ozbiljan razlog da i one ova performanse gledamo s punim poverenjem.

Pre nekoliko godina smo, naime izveno vreme koristili Star štampač ekonomske klase (model NX1000) Bio je to u to vreme jedan od najjeftinijih modela na tržištu. Kupljen je (baš zato što je bio jeftin) kao drugi štampač u kući, da bi sa štedeo dragocenu HP LaserJet Seriesa II+ i poveravanl su mu najprljavlji mogući poslovi se idejom da se u radu potpuno amortizuje Tipična seansa sa njim je bila sledeće: uveče sa pre polaska na spavanje, pusti da štampa bez ikakvog nadzora stedvsalo-trideset strana, a ujutro sva čeka gotove Nakon više od godinu dana intenzivne upotrebe nakon puno kutija perforiranog papira od po 2000 listova i još više potrošenih traka mali Star je još uvek tu u punoj snazi Iza lerlkog plastičnog kućišta krije se jedna veoma žilava tehnologija.

Snimak: Studio Tošić & Nenad Petrović

Ilrzina, pre svega Star ZA-250

Korisna adresa

Sega
11070 Beograd, M. Popovića 6 Sava Center
Tel (011) 222-3579 147 182 222-4323
Fax (011) 455-785 605-579

Olivetti M6-460, 486 DX2, 66 MHz, 100 MB RAM
intel inside

Olivetti Suprema
Razlika u vrednosti

Linija "M6"- Korak u budućnost

Malo je računara koji mogu u potpunosti integrisati najnovija dostignuća tehnologije i odgovoriti budućim izazovima

OLIVETTI-jev odgovor na ovaj izazov je nova profesionalna linija PC "SUPREMA M6" bazirana na procesorima i486 i Pentium

Audio Integrisani "Sound System" sadrži zvučnik, mikrofon i dodatni hardver integrisan na osnovnoj ploči i omogućava zadavanje više od 50 Windows komandi glasom kao i integraciju zvuka i teksta

Nadgradnja Modeli na bazi procesora i486 mogu se nadgraditi "OverDrive" procesorom na bazi Pentijum tehnologije zahvaljujući "Blue OverDrive Socket"-u integrisanom na osnovnoj ploči

Grafika Linija "M6" je opremljena "Local Bus" arhitekturom Podržava VGA, VGA Plus i SVGA rezoluciju

Sigurnost Sa 11 sigurnosnih opcija na 3 nivoa (Hardware, Software i mehanički) Vaši podaci nemogu biti bolje zaštićeni

Kvalitet Garancija na osnovu 2 500 000 sati testiranja pod različitim uslovima i sertifikata 11 vodećih svetskih standarda za kvalitet

Kompatibilnost Sertifikovana i testirana sa preko 300 aplikacija vodećih svetskih softverskih kompanija

Ergonomičnost Maksimalna prilagodjenost psiho-fizičkim zahtevima korisnika po standardu ISO9241 CEE (low emission, high refresh rates,)

Izbor Za zadovoljenje Vaših zahteva OLIVETTI nudi modele M6 na bazi procesora

- i486 SX - 33MHz
- i486DX - 33MHz
- i486DX2 - 66MHz
- Pentium - 66MHz

OLIVETTI M6 SUPREMA
i486 ili Pentium mikroprocesor
Integrisani audio sistem
Video sa Local Bus-om
Trostruki sigurnosni sistem
Pentium model sa Ethernet LAN i SCSI disk kontrolerima

olivetti

OLIVETTI ENERGODATA, Palmira Toljatija 5, 11070 Novi Beograd, Tel (011) 698 512, 692 007 Fax (011) 695 912

NAJVEĆA NAGRADNA IGRA U ISTORIJI ČASOPISA RAČUNARI

45.000 DEM U HARDVERU I SOFTVERU

JEDINSTVENA PRILIKA DA SPOJITE LEPO SA KORISNIM

I DOĐETE DO JEDNE OD PREKO 100 VREDNIH NAGRADA

OD KUTIJE DISKETA DO 486/50 MHz SA LOKAL BASOM

• Velika nagradna igra u četiri broja „Računara" • Preko 100 nagrada od čega 8 u vrednosti od preko 1000 DEM • Nagradni kupon u svakom broju • Kuponi se šalju na kraju igre • Javno izvlačenje do kraja oktobra • Uručenje nagrada odmah po izvlačenju • Rezultati igre u decembarskom broju

U saradnji sa nekoliko vodećih domaćih firmi za promet i razvoj računarske opreme redakcija „Računara" po tradiciji za ovo doba godine organizuje veliku nagradnu igru za čitaoce koja će trajati čitavog leta. Za razliku od dosadašnjih igara, od vas ovoga puta ne tražimo ništa posebno – izvolite samo da sakupite kupone sa zaštitnim znakovima naših sponzora i fotostavite ih kada za to bude čas, na našu adresu.

Igra će se odvijati sledećim redom.

- ***računari*** 92 COMTRAD kupon
- ***računari*** 93 INFO D kupon
- ***računari*** 94 PC CLUB kupon
- ***računari*** 95 IMTEL kupon
- ***računari*** 97 rezultati i izvlačenj sa izvlačenja

NAGRADE

1. ComTrad Računar 486DX / 50 MHz, hard bus, 8 MB RAM local bus video za 1 MB RAM-a, hard disk 210 MB FDD (5.25" ,3,5") SVGA color (Daewoo) vellko tower kutište, tastatura, miš PLUS inkjet štampač Hewlett Packard DESKJET 500

3. IMTEL Specijalna servisne nagrade garantne pomoć u vrednosti od **5.000 DEM** za kupovinu opreme po sapetvenom izboru

3. Info D Računar 486DX / 50 MHz, IBA 4 MB RAM, Trident 1 MB hard disk 210 MB FDD (5,25" 3,5") SVGA color (Frontek), vellko tower kutište, tastatura, miš

6. PC Plus Računar 386DX / 40 MHz, 4 MB RAM VGA 512 KB hard disk 120 MB (Seagate ST3144A) mona VGA monitor (RIC) FDD (5,25" 3,5") fax/modem kartica, SoundBlaster „Zolrik" za zvučnicima

6. Jugodata Računar 386DX 40 MHz, Trident (8900), 4 MB RAM, hard disk 80 MB mona VGA (8CAN) mini tower kutište tastatura miš

6. Computer Laserski štampač Hewlett Packard LaserJet 4L

7. OGA Video VGA kartica (Truevision 16P PAL) (OGA) PAL grafička maška za konverziju VGA u video sa rezolucije 800x600 u 256 boja i VGA prikaz do 1024x768 u 256 boja. Omogućuje snimanje VGA grafike na video, kao i njeno spajanje sa živom slikom

8. LS Data Džepni kompjuter PSION series 3, 256 KB RAM.

PROPOZICIJE

1. Pravo učešća i mogućnost dobijanja neke od nagrada imaju svi čitaoci „Računara" U nagradnoj igri ne mogu da učestvuju lica zaposlena u firmama sponzora i u „Računarima" odnosno BIGZ-u, članovi njihovih porodica, kao ni lica koja su redovni saradnici organizatora igre.
2. Nagradni kuponi objavljuju se u četiri broja u „Računarima" broj 92 (maj 1993.), broj 93 (jun-avgust 1993.) broj 94 (septembar 1993.) i broj 95 (oktobar 1993.)
3. Učesnici treba da pošalju sva četiri nagradna kupona. Za dodelu nagrada u obzir će se uzimati jedino originalni nagradni kuponi iz „Računara"
4. Sva četiri nagradna kupona treba – ispljuštene zajedno – dostaviti na adresu „RAČUNARI" Bulevar Vojvode Mišića 17, 11000 Beograd, sa obaveznim naznakom NAGRADNA IGRA. Rok za dostavu 28. oktobar (uvažava se poštanski žig)
5. Kuponi mogu da se šalju isključivo u standardnim koverlama manjeg ili srednjeg formata. U obzir za dodelu nagrada neće se uzimati veliki koverti (formata A-3 ili veći) koverti prevlđeni u viseći mišić, paketići i slično.
6. Redakcija „Računara" i sponzori kao organizatori na mogu da snose nikakvu odgovornost za eventualni gubitak, oštećenje ili kašnjenje vašeg pisma u pošti.
7. Izvlačenje imena dobitnika nagrada izvršiće se javno nasumičnim izborom. O tačnom datumu i mestu izvlačenja učesnike će obavestiti preko sistema za međunarodne komunikacije Sezam.
8. Nagrade će biti uručene dobitnicima odmah po izvlačenju
9. Kompletni rezultati izvlačenja će biti objavljeni u decembarskom broju (97) „Računara"

9. Admeom Štampač PANASONIC KX-P1123 (24-igličnl)
10. SoftProjekt Štampač PANASONIC KX-P1123 (24-igličnl)
11. Rekmarl Štampač CITIZEN 124 D (24-igličnl)
12. SAGA Štampač Star LC 20 (9-igličnl)
13. MR Sistem Štampač HYNDAI HDP-800 (A3 9-igličnl)
14. IBC Štampač HYNDAI HDP-800 (A3 9-igličnl)
15. ASYS com. Štampač EPSON LX 186 (9-igličnl)
15. MICRO AM Grafička kartica DIAMOND Stealth 24 (16,7 miliona boja) SE grafički čip na 1280x1024 sa 1 MB RAM i prateći softver za podršku velikog broja aplikacija
17. MICRO AM Grafička kartica DIAMOND SpeedSTAR, HI color (32,000 boja) TSENG grafički čip na 1664x768 sa 1 MB RAM i prateći softver za podršku velikog broja aplikacija

18. Mikrodata On-line UPS uređaj od 600 VA za potpuno automatsko upravljanje
19. KR DATA UPS – 750 nagradna nagrađenje za dodatnu računara 750 W
20. MP Biro Color VGA monitor (DTK, M")
21. Telekom Panasonic telefon
22. Iro-Coop Plenić de

23. Softverski paketi:

- dBASE IV (Sprint)
- Texte Pascal for Windows (Sprint)
- Turbo Pascal 6.0 – DOS ver (Sprint)
- Quatro pro 4.0 (Sprint)
- Autodesk – CA LAB (dva kom.) (OBA)
- Autodesk – CHAOS (dva kom.) (OBA)
- MS DOS 6.0 Upgrade (MSoft)
- Borland Quatro pro 1.0 (Softland)

26. Kursevi:

- Kurs po izboru (za deset dobitnika) (Tehnicom)
- AUTOCAD ili CORIEL DRAW (za bost dobitnika) (Mlaro Anima)
- Kurs po izboru (za pet dobitnika) (Institut "Boris Kidrič")

26. Miševi

5 komada – Sprint
2 komad – Tehnicom
1 komad – MP Biro

26. Diskete

6 kutija SAGA
2 kutije – Tehnicom

27. Filteri za monitor

5 komada – Sprint

28. Knjige

10 knjiga – Institut „Boris Kidrič"
1 knjiga – CET

29. Držač za papir i ribon – CET)

30. Podsetnik za korišćenje tastature za WP 5.1 i Windows – CET)

31. Pretplata za Soncm

- 1 godišnje
- 2 polugodišnje
- 3 trimesečne
- 4 jednomesečne

28. Pretplata za časopis Rekmarl

- 2 godišnje
- 2 polugodišnje
- 2 trimesečne

NOVI SPONZORI

COMPUTER
Knjaža, Beograd, Prva Subotica 10/11, Tel. (011) 711-916, 711-967, Fax: (011) 711-478

Osnovna delatnost firme COMPUTER iz Beograda je prodaja i servisiranje PC računara. U njihovom programu je i prodaja programa kompanije DEC, ali i dodatne opreme i raznog potrošnog materijala.

ITRO-COOP
Antje. M.Vujovića 6, Tel. (081) 881-488, Fax: (081) 864-256

ITRO-COOP proizvede za istraživanje, organizaciju, proizvodnju i promet je osnovao Dragomun Jovanović 1988. godine. Osnovna delatnost je proizvodnja konzervirajućeg nanošenja, što podrazumeva razvoj novih proizvoda i tehnologija, organizovanje proizvodnje, informacije tržišta kao i promet za veliko i malo.

NAGRADNI KUPON

***računari* 94**
Sakupljaju kupon do kraja nagradne igre

3

SVI NAŠI SPONZORI

COMTRAD

Genex apartmani, V. Popovića 6, Beograd, tel: 222-41-61 222-36-62, fax 222-41-36

COMTRAD Group je organizacija osnovana u Venecueli 1984 godine i bavi se proizvodnjom i prodajom informatičke opreme i usluga. Zapošljava više od 700 ljudi u oka 40 firmi širom sveta. Kompanija COMTRAD YU sa sedištem u Beogradu, pored jugoslovenskog zadužena je i za istočno-evropsko tržište a deo posla preuzima i sestrinska firma „SITI". Kao deo grupe COMTRAD YU nudi PC AT računare 386SX 386 i 486 „Epson" i „Hewlett Packard" štampače „Novell" mrežni softver multipartne kartice za UNIX, UPS uređaje mrežnu opremu kao i sve vrste prateće opreme

INFO-D

Gradski park B.B. (Zgrada Crvenog krsta), 11000 Zemun, tel./fax: 613-832.

Agencija za informatički inženjering marketing i premet INFO-D je osnovana 1989 godine. Bavi se razvojem softvera kao i prodajom računara i pratećeg hardvera. Nudi komponente najpoznatijih svetskih proizvođača Western Digital Conner Intel Teac, Toshiba Epson. U saradnji sa domaćim firmama INFO-D izrađuje standardni i namenski softver a razvijen je i korisnički alat za domaće programere

IMTEL

Bulevar Lenjina 184b, Novi Beograd, tel: 136-430, fax 136-[illegible]

IMTEL, Institut za mikrotalasnu tehniku i elektroniku, je naučno-istraživačka kuća osnovana 1973 godine IMTEL u svom računarskom tržišno orijentisanom programu nudi proizvodnju prodaju i servisiranje računara i računarske opreme prodaju projektovanje instalaciju i održavanje računarskih mreža izradu programa vezanih za poslovanje firmi i preduzeća, računarski sistem za bespapirno poslovanje *IMTE Loffice* obuku – počevši od osnova računara do usko specijalizovanih rogramskih paketa te stručne konsultacije. Preko javne informacione elektronske mreže IMTELBBS svaki korisnik IMTEL-ove računarske opreme može da pristupi IMTELNET-u koji je povezan sa JUPAK-om

PC PLUS

Budimska 16a, 11000 Beograd, tel/fax 337 130, 349-227

Firma PC PLUS je osnovana 1992. godine. Osnovna delatnost PC PLUS-a je projektovanje i instalacija računarskih mreža zasnovanih na Novell operativnom sistemu i programu Microsoft Windows for Workgroups kao i izrada računarskih sistema svih konfiguracija sa pratećom opremom po želji korisnika. PC PLUS vrši kompletno servisiranje sistema i opreme. Kao ovlašćena firma za prodaju originalnog softvera najpoznatijih proizvođača, PC PLUS isporučuje najnovije programske pakete koji se trenutno nude i u svetu

JUGODATA

Bulevar revolucije 305, 11000 Beograd, tel/fax: 610-306, 419-700, (081) 11-618.

JUGODATA projektuje računarske sistema po zatevu kupca. Isporučuje svu opremu od najskromnijeg PC računara do mreže jakih računara i neophodne periferijske opreme, oprema sistemskim softverom iz svoje standardne ponude ili ga izrađuje po zahtevu kupca. Obezbeđuje obuku te servisiranje i održavanje računarskih sistema. Kao generalni zastupnik singapurske firme „S&C Computer" JUGODATA nudi i pojedinačnu prodaju računarskih komponenti periferne opreme i potrošnog materijala. Ima brojne instalacije lokalnih mreža baziranih na Novell Netware sistemu i razvija aplikacije za svakva višekorisnička okruženja

MICRO DATA

Ustanička 64/XB, Beograd, tel/fax 464-801 468-550

Preduzeće MICRODATA se bavi prodajom svih konfiguracija PC računara, sistema za neprekidno napajanje (UPS) "Epson" štampača i mikro-filmske opreme. Izrađuje softver za finansijsko poslovanje i knjigovodstvo ta vrši projektovanje računarskih mreža i kompletno servisiranje računarskih sistema.

O S A

Šumadijski trg 6a, Beograd tel 021-892, fax 630-845.

Osnovna delatnost je profesionalne računarska grafike u oblasti projektovanje na računaru (CAD) fotorealistična vizuelizacija prostornih modela i računarska animacija. Korisnicima računara OSA nudi savetovanje projektovanje sistema isporuku i umrežavanje računara, periferijske uređaje standardni ili specijalni softver kao i obuku kadrova i održavanje sistema

S A G A

Sage Milentija Popovića 9 (Seva Center), 11070 Novi Beograd, tel/fax 222-35-79, 147-122, 222-43-23/lok 266, 289.

Osnovna delatnost SAGE je kompjuterski inženjering – proizvodnja projektovanje i instalacija računarskih sistema i mreža i druge prateće opreme SAGA je zastupnik američke kompanije ATHANA japanske firme STAR i američke firm „American Power Conversion" (APC)

ANIMA

29. novembra 71, Beograd, tel: 348-940, 343-294.

ANIMA je preduzeće za računarski podržano arhitektonsko projektovanje predstavu i animaciju. Nudi konsalting iz oblasti nabavke i primena savremene računarske opreme u arhitektonskoj praksi kao i obuku kadrova. Osim toga ANIMA nudi iznajmljivanje visokokvalitetnog HP plotera laserskog i kolor štampača kolor skenera i slajd rekordera, kao i prenošenje slike se računara na video traku

CET

Španska 45, Beograd, tel/fax: 340-043.

Preduzeće CET je osnovano pre 3 godine i bavi se prodajom računara i različita računarske oprema – hardvera, licencnog softvera, potrošnog materijala strane i domaće literature kompjuterskog nameštaja. CET je ovlašćeni distributer više softverskih firmi među kojima su i Borland i Microsoft

SOFT PROJEKT

Vranešl, Vrnjačka Banja, tel 036/61-847, fax: 036/67-271

Preduzeće za informatički inženjering i trgovinu SOFT PROJEKT iz Vrnjačke Banje postoji već 5 godina. Osnovne delatnosti je izrada softvera po narudžbini i specijalnost knjigovodstveni softver koji se realizuje pod DOS-om i NOVELL-om. Osim toga SOFT PROJEKT je i distributer računara i prateće opreme tako da na tržištu nastupa se ponudom po principu „ključ u ruke"

ASYS COMMERCE

Bulevar 23. oktobra 25, Novi Sad, tel. 021/623-020, 624-591 fax 021/615-867

ASYS COMMERCE je preduzeće za računarski inženjering i konsalting. U svojoj ponudi ima računarsku opremu raznih performansi i kapaciteta i pesebno štampače (Epson) linijske štampače (Dataproducts) i uređaje za neprekidno napajanje (UPS). ASYS COMMERCE posvećuje posebnu pažnju razvoju softverske servisne službe

SPRINT

Poslovni centar "Apolo" II sprat, Trg slobode 3, Novi Sad, tel. 613-800, 611-605, 623-717 fax 623-901

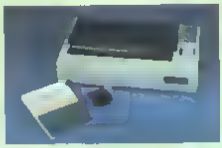
SPRINT INFORMACIONI SISTEMI je asocijacije zenatskih radnji koje su se integrisale pre četiri i po godine. Bavi se sklapanjem održavanjem i servisira njem računara projektovanjem informacionih sistema projektovanjem i izradom računarskih mreža razvojem aplikativnog softvera nabavkom originalnog softvera i obukom korisnika.

INSTITUT „VINČA"

Centar za permanentno obrazovanje

Nemanjina 4/10, Beograd, tel: 659-300, 659-466, 641 155/lok 137 161 fax 662-486

Bavi se obrazovnom i izdavačkom delatnošću u oblasti programiranje programskih jezika i primene računara, kao i izradom poslovnih i poslovno-tehničkih informacionih sistema. Jedini je autorizovani obrazovni centar u Srbiji za proizvode firmi SCO ORACLE BORLAND AUTODESK i LOTUS



ISC

ISC Ups 28/17 Cerak, Beograd, tel/fax 606-785.

Preduzeće ISC se bavi konsaltingom u oblasti informacionih sistema i prodajom IBM-kompatibilne opreme - računara AT/386 i 486 (ISA EISA VL Bus) se orijentacijom na jednoprocesorske i višeprocesorske servere grafičke stanice i mrežna terminale

MP – BIRO

Seogradska 41 Beograd, tel 341-302, 496-361

MP BIRO je mešovito grčko-jugoslovenko preduzeće za inženjering i marketing u oblasti računarske tehnike i informatike odnosno biro za organizaciono-tehnološke poslove, automatsku obradu podataka stono izdavaštvo i prodaju računara i prateće opreme. Za tri godine postojanja, MP BIRO je razvio svoju servisnu mežu u Beogradu Nišu Podgorici Prištini

MICRO-AIR

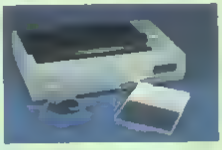
29. novembra 71 Beograd, tel: 343-940, 343-294

Preduzeće MICRO-AIR se bavi proizvodnjom računara se CAD i grafičke aplikacije u raznim konfiguracijama. Zastupnik je nekoliko poznatih svetskih firmi i nudi njihovu računarsku opremu „Hewlett-Packard" štampače skenere plotere i dodatnu opremu „Epson" štampače EIZO profesionalne monitore „Diamond" i „Hercules" grafičke kartle „Polaroid" film-rekordere

ADACOM

Čika Ljubina 12, Beograd, tel 625-289, fax 629- 672.

ADACOM se bavi sklapanjem i prodajom IBM kompatibilnih računara od klase 386SX do 486 na 50MHz EISA. Osim toga nudi i svu prateću opremu za računare – štampače miševe, strimere i drugo. ADACOM obavlja i projektovanja i instalaciju računarskih mreža, za koje pruža potpunu softversku i hardversku podršku



MR SYSTEMS

MR Sistem Lenina 30, Beograd, tel: 659-005, 606-785, fax: 021-385

MR SYSTEMS je preduzeće se razvoj i izgradnju informacionih sistema, koje posluje u zemlji i inostranstvu. Osnovne delatnost je inženjering u oblasti informacionih sistema se poslevnu i industrijsku primenu na UNIX i DOS platformi u računarske opreme okuženju se akcentom na CIM rešenjima

KR DATA

Knezevića 18a, Novi Sad, tel 021/623-611, 611-163, fax 021/26-322

Glavna delatnost preduzeća KR DATE je projektovanje i izrada uređaja se neprekidno napajanje (KR UPS) kao i projektovanje izrada instalacija i održavanje profesionalnih sistema se ozvučavanje KR DATA se bavi i prodajom servisiranjem i održavanjem računara i računarske opreme popravkom raznih elektronskih uređaja.

LS DATA

Braćenička 2/1 tel/fax: 626-780, centrala 633-066/18

LS DATA distribuira PSION računare i pruža kompletan inženjering – izradu projekata, instalaciju PSION opreme, povezivanje PSION opreme se PC računarime izradu programa se PSION i PC-računara kao i obuku i servisiranje LS DATA prodaje i PC računare štampače i ostalu računarsku opremu

TEHNIKOM

Vojvode Mišića 37 Beograd, tel/fax: 685-11-00, tel 650-524/1200.

Preduzeće TEHNIKOM je osnovano 1991 godine i bavi se prodajom računare servisnim uslugama u oblasti elektronike izradom softvera, projektovanjem a posebno obukom. TEHNIKOM ima sopstveni školski centar sa 6 polaznika u kojem organizuje kurseve ze MS-DOS korisničke programe, Novell i Unix operativne sisteme

SOFTVER

Baze podataka

Borland Paradox SQL: Bolja veza sa različitim formatima

Novost koja dolazi iz Borlanda je *Paradox SQL (structured query language) Link* za *IBM Distributed Data Connection Service* (DDCS/2 – povezivanje distribuiranih podataka) koji omogućava bolji pristup podacima iz različitih baza podataka.

Kako navodi Borland IBM veza nudi ukupno osam prikljućaka za *Paradox 4.0*, program koji trenutno pristupa podacima na DB2 SQL/DS SQL/400 i Database Manager (DBM) serverskim bazama podataka.

DDCS/2 je deo IBM-ove arhitektura za distributivna relaciona baze podataka (DRDA – *Distributed Relational Database Architecture*) koja omogućava međusobno povezivanje sistema za upravljanje relacionim bazama podataka. DRDA podržave SQL kao standardni interfejs za pristup distribuiranim relacionim podacima.

Po rečima Borlanda, SQL Link nudi visoke performanse bezbednost, i transparentnost pri pristupu podacima smeštenim na različitim SQL serverskim bazama podataka. Osim toga, obezbeđuje automatsko prevođenje *Paradox QBE* i meni-komandi u odgovarajući SQL dijalekt. Izvorne SQL naredbe mogu biti ugrađene u programe pisane u PAL-u i tako se prenositi bazi podataka. Uz pomoć SQL prenosa, programeri imaju sasvim otvoren pristup svim funkcijama serverske baze podataka, uključujući i procedure trigera i rečnike podataka.

IBM ističe da će odluka Borlanda da primeni DRDA arhitekturu učiniti računarsku snagu daleko dostupnijom Paradox SQL Link za DDCS/2 nudi prolaz (*gateway*) do različitih baza podataka uz obezbeđen integritet, pouzdanost i mogućnost oporavka podataka, što je izuzetno važno kod aplikacija za kritične primene.

Paradox SQL Link veze podržavaju i *InterBase 3.3 Novell NetWare SQL 3.0 ORACLE 6.0 Server Microsoft/SYBASE SQL Server, IBM OS/2 Database Manager DB2* preko programa *MDI Database Gateway 1.01* kao i DEC-ov VAX Rdb/VMS 4.1

Preporučena cena je 395 USD, a za sve verzije je neophodan i softver *Paradox 4.0* koji se prodaje zasebno po ceni od 795 USD

Mreže

Alatke za upravljanje NetWare mrežom

„Novell" je najavio *NetWare Management System* (NMS) *Software Development Kit* (SDK) verzija 2.0, koja će omogućiti da *NetWare* okruženjem upravlja bilo koji sistem zasnovan na protokolu SNMP (*Simple Network Management Protocol*)

NMS SDK programerima nudi sveobuhvatnu zbirku alatki i otvorene API-je za razvoj upravljačkih uslužnih i aplikacija. Nove verzija ima *NetWare* serversku bazu informacija za upravljanje (*Management Information Base*) što znači da svaka SNMP konzola može da upravlja *NetWare* serverima. SNMP desktop transport omogućava programerima pisanje aplikacija za upravljanje koje informacije sa desktop sistema prenose bilo kom SNMP upravljačkom sistemu.

Predstavnik „Novell"-a ističe da novi SDK omogućava proizvođačima integraciju se NMS konzolom ili bilo kojim SNMP sistemom, da upravljaju *NetWare* operacijama i *NetWare* kijentima iz jednog centra. Osim toga upravljanje *NetWare* serverima i desktop sistemima se može integrisati se upravljanjem ostalim mrežnim uređajima.

NetWare MIB administratorima mreže omogućava da iz bilo kog sistema za upravljanje koji primenjuje SNMP pristupe informacijama kao što su upravljanje konfiguracijom identifikacije problema resursi

NMS SDK programerski interfejsi (API) nude programeru mogućnost da višeslojni softver za upravljanje mrežnim uređajima kao što su *router* i *hub*, integrišu sa upravljačkim aplikacijama koje nudi Novell n svom paketu *NetWare Management System*

Cena od 495 USD podrazumeva besplatnu tehničku pomoć i redovnu automatsku dogradnju

ŠTA IMA NOVO

SOFTVER

Radne tabele

Lotus Improv for Windows: Još uvek na staklenim nogama

Improv for Windows 2.1 novo izdanje Lotusovog programa za radne tabele u kojem je primenjen sasvim novi dizajn omogućava povezivanje sa sa raznim softverom za radna grupe *Lotus Notes* i se ba zama podataka

Originalna verzija *Improv*-a se pojavila u februaru i Mla je namenjena samostalnoj upotrebi Verzija 2 1 sadrži instalacionn proceduru za rad u mreži koje je uvek ista, bilo da program učavsele za samostalan rad za distribuciju u mreži ili za server

Improv instalacione shema je postala standard za celu liniju Lotus proizvoda, tako da je nedavno dodata n 1 2-3 za DOS v 3 4a 1-2-3 za *Windows* v 4..1 *Ami Pro* tekst-processo.. e uskoro će se naći i u ostalim Lotus-ovim proizvodima

Novi *Improv* uma i da čita i da zapisuje WK4 datoteke iz 1 2-3 v 4 za *Windows* Isporučuje se (od 15. juna) sa tehnologijom *Q+E Extend for Improv Starter Edition* preuzetom od kompanije „Q+E Software" koja omogućava korisniku da traži podatke iz raznih eksternih baza podataka i prenosi ih u *-improv* radnu tabeln Sličnn nlogn obavlje Lotusov *Datalens* u paketu 1 2-3 *Q+E Extend for Improv Starter Edition* pruža prtup Borlandovim bazama *Paradox* and *dBASE* le *Btnave* and *Microsoft Excel*

Veza se programom *Notes* se ostvaruje preko *Application Field Exchange* (AFE) tako da se polja (*fields*) iz *Notes*-a povezuju sa *Improv* radnim tabelama Kada se *Improv* tabela promeni nove vrednosti se pomoću OLE tehnike automatski pojavljuju u odgovarajućem *Notes* dokumentu AFE radi I n obrnutom smeru – ažurira polje n *Notes* dokumentu..e promene se pojavljuju n *Improv*-u

Improv for Windows 2.1 će imati verzije na holandskom, francuskom nemačkom italijanskom španskom i švedskom jeziku Preporučena cena je 495 USD

Minimalni zahtevi podrazumevaju računar 386 4 MB RAM tvrdi disk od 12 MB i instaliran *Microsoft Windows 3.1* Podržani su svi važniji mrežni operativni sistemi *Novell Netware 4.0 3.11 i 2.15c IBM PC LAN 1.2 i 1.34xs Banyan VINES 4.0, 4.11 i 5.0 Microsoft LAN Manager 2.1e i 2.2 IBM LAN Server 1.3 2.0 i 3.0 Digital Equipment Pathworks 4.1 i NCR StarGroup 3.5 i 2.1e*

Fontovi

Novi Adobe slovni likovi iz životinjskog sveta

Adobe je nedavno izdao dve nove zbirke fontove – prve je inspirisana životinjskim svetom i još nekim popularnim temama i zove se *Wild Type* dok je druga, *Adobe Type Basics*, namenjena korisnicima koji tek počinju da se bave fontovima

U zbirci *Wild Type* se nalaze slova koje podsećaju na razne zveri likove iz stripa crtanog filma (Critters Blnberg slove inspirisana vožnjom skejt borda, zemljotresom alatima (svako slovo liči na nekn alat) iz originalna Adobe zbirke prenesenog je još pet fontove *Mynad Sketch Mynad Headline Mynad Titl Birch* i *Utopia Headline.*

Adobe Type Basics je zbirka 65 slovnih likove (*typefaces)* za početnike koji žele da imaju raznovrsnn biblioteku visoko-kvalitetnih slovnih likove po pristupačnoj ceni i sadrži *Adobe Type Manager* sa 13 etandardnih slovnih likova 22 iz zbirke *Plus Pack* i 30 *Adobe Type 1* likova slova.

Obe zbirke se već mogu dobiti i za Macintosh i za PC platforme *Wild Things* ima cenu od 60 USD a *Adobe Type Basics* 198 USD Do 31 decembra registrovani korisnici *Adobe Plus Pack*-a mogu kupiti dogradnju za *Type Basics* za 39 dolara.

ŠTA IMA NOVO

TEHNOLOGIJA

Pentium računari

Važno je biti prvi, makar se nešto i prodavalo

Firma „Advanced Logic Research" (ALR) tvrdi da je prva (7 avgusta) počela sa isporukom PC računar zasnovanih na Intelovom mikroprocesoru Pentium Vest je dovoljno zanimljiva za zainteresovane kupce i zabrinjavajuća za konkurenciju ali to još nije sve – ALR je odredio početnu cenu od samo 2 495 USD! U njihovoj novoj seriji se nalaze dve osnovna Pentium računara. Evolution V namenjena individualnim korisnicima i Evolution V-D namenjen tržištu mrežnih servera. Obe mašine imaju ugrađen Pentium na 60 ili 66 MHz, petogodišnju garanciju i sistem za hlađenje sa ventilatorom na samom Pentiumu

Evolution V ima 64-bitni dizajn i 8 MB memorije na matičnoj ploči proširivo do 128 MB. Tajna relativno niske cene se krije u tome što u njih $2.495 nisu uračunate neke od osnovnih komponenti – hard-disk drajv i monitor ali obavešteni tvrde da se ipak radi o fantastično povoljnoj ponudi Jer kompletan sistem – sa tvrdim diskom od 170 MB DOS-om *Windows*-om mišem se VL-bus magistralom i 14-inčnim monitorom – košta još uvek samo 3 595 USD

Evolution V-Q server ima 128-bitni dizajn 10 ESA slotove tri za lokalnn sabirnicu jedan gigabajt osnovne memorije 13 drajvova i košta 4 495 USD

Onaj ko naruči svoju Pentium mašinu od firme ALR može da je očekuje za dve do tri nedelje Međutim broj Pentinm procesora koje Intel isporučuje proizvođačima je za sada ograničen

Konsna adresa ALR tel 714-581-6770 fax 714-581-9240

Mikroprocesori

AMD ima mikrokod za 486

AMD (Advanced Micro Devices) najavljuje tri nova mikroprocesora 486 kompatibilna sa Intelovom linijom ali se mikrokodom za koji se tvrdi da je nastao u AMD-u ipak predstavnici kompanije kažu da su svesni da ih očekuje težak sudski spor sa Intelom ali nastavljaju proizvodnju mikroprocesorskih čipova kompatibilnih se Intelovom linjom

Njihov prvi 40-MHz 486SX, kao i 33 MHz Am486SXLV radi na 3.3 volta Prvi je potpuno (pinto-pin ili „nožica na nožicu") kompatibilan se Intelovim 486SX mikroprocesorom dok je Am486SXLV namenjen nisko-voltnoj primeni tj *laptop* i *notebook* računarima i na odgovara podnožjima za Intelove procesore

AMD ističe da njihovi čipovi nude više bilo da su to bolje performanse ili manje potrošnja struje te da su potpuno kompatibilni se softverskim aplikacijama pisanim za PC čija je ukupana vrednost oko 80 milijardi dolara

Prilikom predstavljanja novih čipova, predstavnici AMD su izjavili da firma smatra da na osnovu ugovora iz 1976 ima puno pravo ne korišćenje Intelovog mikrokoda (instrukcija ugrađenih n mikroprocesorski čip). Upravo je taj ugovor bio predmet žestokog spora između dve kompanije a u stvari se svodi na to da li termin „mikrokompjuter" koji se pominje u ugovoru podrazmeva i mikrokod u mikroprocesorskim čipovima. U ugovoru doslovce stoji da AMD ima pravo da koristi „mikrokod sadržan n Intelovim mikro-kompjuterima i perifetijama koje prodaje Intel" AMD smatra da mikru-kompjuteri podrazmeva i mikrokod, dok Intel tvrdi suprotno Zbog čipova 486 Intel je tužio AMD i pre no što su se pojavili

Bilo kako bilo AMD se nada da će osvojiti 10% potencijalnog 486 tržišta. Nudeći proizvođačima originalne opreme (OEM) redovan izvoz mikroprocesora i povoljniju cenn AMD je uveren da ima šanse na uspeh uprkos Intelovim optužbama. Najnoviji čip 486 košta 185 USD za isporuke od po hiljadu komada.

Konsna adresa Advanced Micro Devices tel 408-749-3310 fax 408-749-3375

DOK SU DRUGI SPAVALI

Početkom juna korisnici računara obradovali su se novoj verziji popularnog arhivara ARJ, koja nudi brži i pouzdaniji rad, viši stepen kompresije i, naravno, brojne nove opcije. Novi ARJ 2.41 predstavljen je kao doatojan takmac čuvenog PKZIP-a 2.04 („Računari 89") – pogledajmo koliko su takve tvrdnje osnovane.

Dejan Ristanović

Istorija ARJ-a je kratka ali i zanimljiva – njegov autor je *Robert K Jung*, četrdeset-šestogodišnji hemičar iz Baltimora (*Baltimore, Maryland USA*) Tokom 1990 godine gospodin Jung je proučio postojeće arhivere pa i tada svakako najpopularniji PKZIP firme *PKWare Inc.* Zaključio je da se može napraviti odlji i brži arhiver i pronuo ne posao – februara 1991 godine predstavljen je ARJ 1.0 Širanje ARJ-a je u mnogome zasnoveno na tome što je autor PKZIP-a predugo „spavao na lovorikama" – dok se od marta 1990 do decembra 1992 godine čekalo ne novu verziju PKZIP-a, pojavilo se nekoliko izdanja ARJ a koja su nudila sve veću i veću kompresiju i tako značajno prevazišla mogućnosti PKZIP-e ARJ je (premda sve vreme u *shareware* izdanju) donosio tolike prihode de je autor nešto intres da napusti prethodni posao i u celosti se posveti razvoju arhivera i podršci njegovim korisnicima Rezultati su brojne verzije ARJ a krunisane verzijom 2 41 koja se pojevila početkom juna i izazvala veliko interesovanje kako korsnika računara u Sjedinjenim Državama i u svatu tako i na Sezamu gde se pojavile svega par sati posle svetske promocije

Novosti ARJ-a 2 41 u odnosu na prethodnu beta-verziju (2 39f) nisu ni previše brojne ni previše značajne uglavnom su ispravljeni neki sitniji bagovi vezani ze opcije **-t1** i **-&** i unepređen program koji obezbađuje registraciju paketa Sliche izmene vršene su i u „međuverzijama" koje su nosile oznake 2 39e 2 39b 2 39c 2 39d i 2 39f – bagovi su čišćeni jedan po jedan i unepređivana je sigumost čitavog paketa. Neznatna unapređenja u brzini rada i stepenu kompresije primećena su u verziji 2 39f, ali je u osnovi ARJ 2 41 „isto" što i ARJ 2 39a Zato ćemo se u ovom prikazu uglavnom baviti razlikama između kod nas najpopularnije verzije 2 20 i najnovijeg ARJ-a 2 41, ne obraćajući pažnju na sitne razlike u međuverzijama koje su poslednjih meseci stizale jedna drugu na Sezamovim diskovima o njima nismo pisali u „Računarima" smatrajući da je mnogo zahvalnije predstaviti gotov proizvod

UPOTREBA ARJ-A

ARJ je po mnogo čemu sličen ostalim *shareware* arhivarima – dokazi u obliku samorasapkujuće EXE datoteke ARJ241 EXE koju treba startovati u praznom direktorijumu dobijajući jako razne fakstualne i binarne fajlove među kojima odminira ARJ EXE sam arhiver koji treba da prekopirate u neki od direktorijuma uključenih u *path* (npr C \SYS ili C \UTIL) Kada vam zatrebe arhiviranje otkucaćete, na primer **ARJ e test * *** i tako će nastati TEST ARJ Raspakivnjem ove ahive (npr **ARJ e test**) dobićete originalne datoteke

Eventualne opcije navode se iza minusa – komanda **ARJ e -r TEST * *** će u TEST ARJ upisati ne samo arhivirane datoteka iz tekućeg kataloga već i one iz potkataloga, potkataloga tih potkataloga i tako dalje Više opcija sa obično navodi uz više minusa **ARJ e -r -jm TEST * *** uz sve prathodno podrazumeva i maksimalnu moguću kompresiju po cenu sporijeg rada

Kao što se vidi ARJ se se aspekta konsničkog interfejsa razlikuje od PKZIP a samo u jednoj stvari umesto odvojenih programa koji obavljaju pojedine operacije (PKZIP za arhiviranje PKUNZIP za raspakivanje, PKZIPFIX za popravljanje oštećenih arhiva, ZIP2EXE za kreiranje samoraspakujućih arhive itd) ARJ „sve" drži u ARJ.EXE – arhive sa kranjau sa **ARJ e**, raspakuju se **ARJ e** (ili **ARJ x**), pratvaraju u semoraspakujuće se **ARJ y** – Sada je spisku dodato i popravljanje arhiva komandom **ARJ -jr** Jedine izvršna datoteka koju dobijate u arhivi ARJ241 EXE (osim ARJ EXE narevno) je REARJ program koji omogućava da sve (ili samo neke) arhive na svom disku pretvorite iz formata drugih arhivera (PKZIP LZH ranije verzije samog ARJ-a, itd) u najnoviji ARJ

GDE SU GRANICE?

Najznačajnija karakteristika arhivera je stepen kompresije pa smo najpre bili radoznali de provenmo koliko je *Robert Jung* uradio ne tom planu Konstuisao sle datoteke na kojima smo pre nekoliko meseci testirali PKZIP 2 04 ASCII tekst dužine oko 730 kilobajta (rukopis jedne knjige plus niz poruke sa Sezama) DBF fajl dužine 9 7 megabajta (podaci o svim firmama u Vojvodini) sliku u TIF formatu dužine 1 1 megabajta (tehnički crtež) i sadržaj jednog "privremenog" direktorijume našeg diska, u kome je bilo 87 fajlove od loga dosta teksta, nešto programa nešlo arhiva (zip i arj) itd, ukupne dužine 1 2 megabajta Arhivera smo testirali pri maksimalnoj, podrazumevenoj i najbržoj kompresiji konsteći „standardan" 386/33/64K *cache* računar sa 4 megabajta RAM-e i brzim *Conner* diskom – verovatno bi polako trebalo preći na testiranje na 80486 računarima ali smo ovako izbegli ponovno merenje za druge arhivere Pri testiranju nismo konstili keš program (niti kontroler sa kešom) premda sulor preporučuje keširanje diska ali i savetuje da sa keširanje diskete ograniči na *read* operaciju

Pn testiranju maksimalne kompresije ARJ-a korišćena je opcije **-jm** Vredi reći de imenovanje ove opcije smatramo Junogovom greškom – kompresijom se inače upravlja opcije **-m** pri čemu **-m1** označava visoku kompresiju i sporiji rad a **-m4** najbržu kompresiju uz najmanji procenat arhivirenja Zato mnogi testirajući ARJ konste **ARJ -m1** smatrajući lo maksimalnom postojećom kompresijom i rangirajući ARJ niže nego što zaslužuje Za opciju **-jm** koje se sasvim lepo mogla zvati **-m0** (trenutno **-m0** znači upis u arhivu bez kompresije) retko ko zna pa je i retko ko koristi Pa ipak, ona je tu i njeni efekti su primetni

Kao što se sa slika 1 2 i 3 vidi ARJ 2 41 je ponudio izvesno unapređenje u odnosu na prethodnu verziju – naročito je napredovale standardna kompresija dok kod maksimalne, moglo bi se reći i nema promena. U svakom slučeju razlike između ARJ-a 2 30 i 2 41 po pitanju stepena kompresije posloje ali nisu baš impresivne, pretazak sa PKZIP e 1 1 na 2 4g predstavlja mnogo spektakularniju promenu Pa ipak, s obzirom da je već ARJ 2.20 po ovom pitanju bio odlji od PKZIP-e 1 1 razlike su se u pnličnoj men „nrvelsale" tako da novi ARJ postiže rezultate sasvim približne PKZIP-ovim na nešim lestovima PKZIP je odje kompnmovao (osim na jednom testu) ali su razlike mizerne u poređenju sa duzinom tejla

Svaka nova generacija arhivere implicire sasvim prirodno pitanje „dokle se može ići", tj gde su teorijske granice arhiviranje podataka Jasno je naime da se prolzvoljna daloteka ne može arhivirati na jedan bajt (made smo videli jedan prvoprilski „zip" koji svaku datoteku arhivira na 1 bajt, „smajli" znak chr(1) jedino ime „malih" probleme pn dearhiviranju), ni na dva ni na 10 A može na polovinu originalne dužine ill čak na Ireičnu Da li bi mogle na četvrtinu osminu, ne 100 bajte? Teorijska istraživanja pokazuju da jos nismo blizu granice – profesor Milan Božić sa Matemaličkog fakultata je ne Sezamu ispričao zanimljivu anegdotu o grupi studenata kojima je kao samestralni rad deo pisanje arhivera Bez posebnog truda i pnmenom sasvim klasičnih algontama postignule su kompresije odlje od tadašnjeg ZIP-e i ARJ-a Oduševljenje je medutim bilo kratkog veka pošto se pokazalo da su tako nastali arhiven nesrazmerno sporiji od „pravih" – očigledno je da se pri dizajniranju mora uvek praviti kompromis izmedu dužine datoteke i vremena arhiviranja i da su se autori ZIP-e i ARJ-a pre svega okranuli vremenu, pošto bi retko ko arhivirao podalke ako bi ze sitne operacije bilo potrebno više minuta ili recimo pola sata Izgleda da će se i sledeći godina ZIP ARJ i (možde) drugi arhiveri nadmetali u stepenu kompresije na sve brzim i bržim računenma a mi camo se još dugo pitali „koliko se još može ići" Može još dosta!

Kede već pomenusmo vreme arhiviranja, slike 4 i 5 pokazuju da je ARJ 2 41 i po tom pitanju napredovao ali se nije ni približio PKZIP-u 2 04g Stver ovoge puta nije samo u algontmime PKZIP konsti specifičnosti mikroprocesora 80386/80486, EMS i XMS memoriju DPMI i, uopšte, sve ono što sa na jednom savremenom PC ju „zalekio" ARJ je ograničen na DOS i iz njege izvlači najviše što se moglo izvući – možda je Jo dobro sa aspekta sigumosti (mnogi problemi PKZIP-a 2 04c proslicali su iz raznih sitnih hardversko-softverskih nekompalibilnosti) ali se loše odražava ne brzinu! *Jung* nije smatrao za potrebno de ugradi bilo šta od tih „novotanja" – jedino je otklonjen neprijatni bag koji je otezavao procesiranje lista * * na mrežame koje rade pod *Novell*-om

U ZNAKU FLEKSIBILNOSTI

Možda najjači ARJ-ov argument je fleksibilnost koje nije viđena u svetu arhivera – postoji toliko opcija da je teško i zamisliti stvar koja se ne može uraditi. ARJ, na primer, omogućava preskakivanje datoteka u arhivama, ujedinjenje dve arhive primenu neke DOS komande na svaku (ili samo neke) arhiviranu datoteku promenu redosleda datoteka u arhivi (priložen je ARJSORT.BAT koji obezbeđuje sortiranje datoteka u arhivi proizvoljnim redosledom, nešto kao poznati *Norton*-ov *Directory Sort*, DS), poređenje arhiviranih datoteka sa datotekama na disku, itd. – sve to bez raspakivanja arhivel Fleksibilnost ARJ a naročito se ispoljava pri radu sa tekućevnim višedelnim arhivama.

Arhive se, naime, najčešće prave da bi se neki podaci preneli na diskete i tamo čuvali za buduću upotrebu. Kapacitet diskete je višestruko manji od kapaciteta diska, pa se sasvim opravdano postavlja pitanje šta da se radi ako dužina arhive prevazide 1.44 (ili 1.2) megabajta. Arhive bi se lako i svaki drugi fajl mogle „seći" na delove primenom nekog specijalnog programa ili u krajnjoj liniji, korišćenjem komande BACKUP (MSBACKUP ne DOS u 6.0) ali tako nastaju delovi koji sami po sebi nisu arhive i koji se moraju „nadovezati" na disku da bi se dobila arhiva sa kojom se uopšte može raditi. *Robert Jung* se doselio da mogućnost podele ugradi u sam program i tako su nastale višedelne arhive – ne prvoj disketi je recimo, BACKUP.ARJ na drugoj BACKUP.A01 na trećoj BACKUP.A02 i tako dalje (reklo bi se da se na ovaj način može procesirati najviše 100 disketa, all je u stvari broj znatno veći – može se početi od BACKUP.000 (u kom slučaju BACKUP.ARJ ne postoji) i ići do BACKUP.999). Svake od tih arhiva sadržaće neke delotoeke koje se po potrebi mogu raspakovati i bez pregleda ostalih arhiva – izuzetak su datoteke na kraju arhive koje su silom prilika podeljene na dva dela tako se primenom opcije **-w** može zahtevati da se datoteke u arhivama ako je ikako moguće, ne dele tj. da se svaki deo arhive završava „čitavom" datotekom a ne njenim delom. Nema naravno potrebe da se svaka arhive raspakuje posebno – uvek se sa **ARJ e -v A IME** mogu raspakovati sve zajedno uz naizmenično umetanje disketa.

Arhiviranje na diskete nije jedine prilika u kojoj su potrebne višedelne arhive. Na Sezamu se, na primer, trudimo da rešimo problem visenika modema od 1200 bps a da lpak ne povredimo preva autora *shareware* programe koji zahtevaju da se njihovo delo distribuira kompletno. Zato kompletnu arhivu još jednom arhiviramo ne da bismo je skratili nego da bismo je time podelili na dva ili više delove. Ova operacija se, prirodno obavlja bez posredovanja disketa, odnosno sve rezultujuće arhive ostaju na disku. PKZIP to uopšte ne može da uradi dok ARJ obavlja zadatak sa lakoćom – čak se može precizno upravljati dužinama dobijenih delova.

Velika prednost nad PKZIP-om je i mogućnost nastavke arhiviranja – ako iz nekog razloga ne uspe upis na dvadeset petu disketu neća biti potrebno da ponovo snimate prve dvadeset četiri!

Problem izmena višedelnih arhiva i dalje je akutan – kod PKZIP-a i ranijih verzija ARJ-a izmene su bile potpuno nemoguće a sada je obebeđena opcija **-hu**, kojom se može pokušati dodavanje ili brisanje datoteka u višedelnim arhivama. Ipak, sam autor ARJ-a tvrdi da je to veoma opasne operacija u koju treba da se upuštate samo ako tačno znate šta radite. Ii ako bnšete ii zamenjujete neku datoteku s bm de se dužine arhive bitnije ne promeni. Višedelne arhive dakle i dalje ostaju *read-only* materijal, eli je kod ARJ-a 2.41 ipak učinjen prvi korak ka njihovom revnopravnijem tretmanu.

Fleksibilnost ARJ-a primećuje se i kod kontrolnih pitanja. Keda se, na primer, pri dearhiviranju pojavi datoteka čije je ime jednako imenu postojećeg tajla na disku, ispisuje se kontrolno pitanje koje oslm tradicionalnog Y (obrisati) i N (preskočiti) obuhvata i druge odgovore. G na primer, znači da datoteku treba obrisati i da se sva buduca pitanja treba automatski odgovarali sa *Yes*. A takođe prepisuje datoteku i ubuduće odgovara sa *Yes* na pitanja **Iste klase** (**G** podrazumeva odgovor *Yes* na sve pitanja), S podrazumeva odgovor *No* na slična pitanja. **C** omogućava zadavanje DOS komande i povratak na isti *prompt* po njenom izvršavanju (se TYPE, na primer, pogledemo datoteku i rešimo treba li je brisati ili ne). Sve ove varijanta su postojale i ranije, ali su u verziji 2.41 sistematizovane na nivou čitavog ARJ-a i pojavljuju se kod svih pitanja kod kojih neka od opisanih odluka ima smisla. Postoji čak i neinteraktivni procesor kničnih grašaka (aktivira se opcijom **-&**) koji će dati podrazumevani odgovor na pitanja tipa „Abort, Retry, Fail" u normalnim prilikama svako od tih pitanja izazvalo bi neplanirani prekid rada.

Ipak, sve te lepote imaju i svoju tamnu stranu – uvedeni su silni parametri i podparametri koje će popamtiti valjda samo onaj ko i danju i noću konsti ARJ. Za prosečnog korisnika je često mnogo manji napor da raspakuje arhivu uradi što ima, ponovo je spakuje nego da se pati sa opcijama koje će mu ušleđeti neku minut vremene ali zato i ozbiljno opteretiti pamćenje. Evo kako to izgleda na primeru arhiviranja čitavog sadržaja diska na diskete (*backup*) u dokumenlecuji se preporučuje komanda **ARJ e A backup -r -vvas -a1 -b2 -js -jt -wC** \ m3 PKZIP ce uradili sličan posao uz jednostavno **PKZIP -&P A BACKUP C** *.* Mnoštvo opcija na sreću ne morate baš uvek pamtiti – one koje uglavnom koristite možete da „fiksirate" u SET promenljivu ARJ_SW e možete i kreirati datoteku ARJ.CFG sa nizom podrazumevanih *switch*-eva u više redova.

U večitoj dilemi projektanata softvera koji uvek moraju da prave kompromis između fleksibilnosti i jednostavnosti kod autora ARJ-a preovladala je težnje da program bude mocan i fleksibilan po cenu uvođenja nekih opcija koje će zaista malo ko zapamtiti i koristiti.

STARO I NOVO

ARJ 2.41 je doneo brojne sitne novilete koji se odnose kako na rad sa višedelnim arhivama tako i na mnoge druge operacije. Izdvojili bismo nešto što je do sada bilo privilegija vlasnika laserskih (WORM – *Write Once Read Many times*) diskova: čuvanje više verzije iste datoteke. Često je, naročito kod programerskih poslova, pogodno sačuvati više prethodnih verzija kako biste se, ako se ispostavi da je neki novi koncepl doneo više šteta nego koristi, mogli vratiti prvobitnom rešenju. To se radi u okviru lakozvanih *backup* arhiva odnosno dodavanjem opcije **-jb**.

Primetili smo i poboljšanja mehanizma samoraspakujućih EXE datoteka. ARJ je naime „oduvek" mogao da kreira i nezavisni EXE fajl koji ca sa automatski raspakivati po startovanju (pogodan način da program pošaljete nekome ko nema pn ruci ovaj arhiver ili čak ne zna kako da ga konsti) ali se sada to obavlja na dva nivoa – „pravi" samoraspakujući EXE je za oko 17 kilobajta duži od arhive ali zato podrzeva većinu mogućnosti ARJ-a dok „mali" EXE dodaje samo 15 K ali zato ne obezbeđuje zaštitni omotač i neke druge komendne opcije (šampion po ovom pitanju i dalje ostaje LZH čiji je EXE samo kilobajt-dva duži od polazne arhive). Samoraspakujuća EXE arhiva kleira se komandom **ARJ y -je ime** gde je *ime* naziv već kreiranog ARJ fajla. Ako želite manji EXE koji pn raspakivanju obezbeđuje manje komlore koristite **ARJ y -je1 ime**. Čak i tom manjem EXE tajlu (zvanom *AnSfuJi*) sa da su dodate neke opcija npr. raspakivanje datoteke u razne direktorijume, izdvajanje novih datoteka iz arhive i lome slično. Najveće ubrzanje novoga ARJ a primetili smo baš kod samoraspakujućih arhiva smeštenih na diskelame.

Mada mogućnost „višejezičkog" rada za arhiver nije naročito bitna, *Jung* se potrudio da podrži mehanizam kodnih stranica predstavljen u DOS-u 3.30. Sada se u imenima arhiva komentarima i na drugim mestima mogu naći i strane slova a da to ne stvara dvoznačnost brojnih ARJ-ovih opcija. Naravno pretpostavlja se da je konsnik rasporedio znake preme nekoj od zvaničnih *Microsoft*-ovih kodnih stranica.

	TEKST ASCII tekst dužine 729237	DATOTEKA DBF datoteka dužine 9679447	SLIKA TIF slika duž. 1060166	RAZNO 87 fajlova duž. 1251105
ZIP max 2.04 st. min raspak	0:27.0 296805 0:19.0 299351 0:09.0 344618 0:05.0	4:04.0 1081071 1:45.5 1141269 0:55.5 1515069 0:39.0	0:29.0 39482 0:11.0 41057 0:05.5 54277 0:05.0	0:25.5 517021 0:26.0 520088 0:15.5 563999 0:19.5
©IP max 1.10 st. min raspak	0:21.0 323068 0:21.0 323068 0:07.5 361320 0:05.0	7:55.5 1368093 7:55.5 1368093 1:02.5 1773233 0:39.0	0:55.5 54162 0:55.5 54162 0:06.0 44588 0:05.0	0:35.5 554119 0:35.5 554119 0:14.5 630125 0:10.0
ARJ max 2.30 st. min raspak	0:34.0 301757 0:25.0 306223 0:13.0 364176 0:08.0	11:37.0 1130529 3:37.0 1224808 1:37.5 1646040 0:45.5	2:10.5 40223 0:24.0 45263 0:09.5 59480 0:05.0	1:10.0 517901 0:44.5 522605 0:25.5 587530 0:24.0
ARJ max 2.41 st. min raspak	0:27.0 301027 0:24.5 301137 0:11.2 356756 0:06.5	7:48.3 1127415 3:21.8 1178649 1:19.7 1591098 0:42.9	0:31.5 40183 0:25.3 44470 0:07.6 50526 0:05.1	0:54.2 517410 0:36.2 518565 0:21.1 581966 0:21.0

Slika 1 – Numerički parametri vremena erhiviranja i dužine datoteka nastalih erhiviranjem raznih tipova podataka

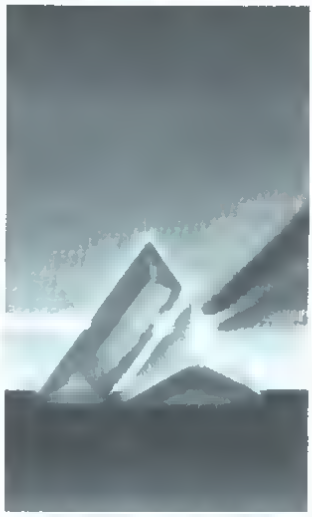
– YU slova po YUSCII rasporedu ne mogu se razlikovati od DOS-ovih komandnih znakova pa su prema tome i dalje nepoželjna u imenima datoteka

Od „sitnih" novosti pomenimo opciju **-hc** koja omogućava zadavanje DOS komande koje će se izvršiti pri samom startovanju ARJ-a (npr. **-hcDIR** = ARJ), kao i opciju **-jd** kojom se raspakuju samo one datoteke iz arhive koje mogu da stanu na disk. Pri korišćenju na već komprimovanom disku (DBLSPACE, *Stacker*, itd.), ova opcija se ne preporučuje

Jedan od ozbiljnih nedostataka ARJ-a uvek su bili memorijski zahtevi – za arhiviranje datoteka potrebno je obezbediti oko 300 kilobajta slobodne memorije, dok za raspakivanje treba „samo" 175 kilobajta. Kada program koristite iz „čistog" DOS-a, ovaj podatak nije od posebnog značaja, ali se arhiven često pozivaju i iz sopstvenih programa, recimo iz neke *Clipper* aplikacije. Da bi se napravila rezervna kopija podataka na disketama (*backup*). Tada je raspoložive memorije malo pa je 300 kilobajta neprijatne cifra, znatno veća od one koju zahteveju konkurentski arhiveri, na Sezamu recimo, još nismo uspeli da instaliramo ARJ kao arhiver pomoću kojeg se može komprimovati korisnikov *pad*. Iako autor tvrdi da je u verziji 2.41 učinio izvesne korake da bi smanjio memorijske zahteve, ne bi se reklo da je taj trud dao prevelike rezultate – ukoliko se arhivirayu datoteke iz poddirektorijume sa dugim imenima, događa se da ni 300 kilobajta nije dovoljno za uspešan završetak operacije

POUZDANOST NA VIŠINI

Ključno pitanje pri vrednovanju arhivera je pouzdanost – da li će arhiva zaista moći da se raspakuje, da li će se time dobiti originalni fajlovi i šta se može učiniti ako je neki od njih oštećen. Oštećenje arhive nastaje iz dve razloga – zbog nepouzdanosti samog arhivera, odnosno zbog nekog njegovog bage ili fizičkog oštećenje datoteke. ARJ je ranije imao nekih problemčića i bagova koji su pod određenim okolnostima mogli da proizvedu nekorektiu arhivu – naročito je po zlu bilo poznato arhivirenje na diskete u prisutnosti nekog *cache* programa. Reklo bi se da su ti problemi sada definitivno otklonjeni – za prvih mesec dana testiranja niko nije imao problema sa loše kreiranim arhivama. Oštećenje datoteke je međutim nešto što se ne može izbeći – delovanje virusa, loš sektor na disku, greška pri pranosu



modemom ili, najčešće, snimanje na neispravnu disketu, proizvešće arhivu čijim se raspakivanjem ne mogu dobiti originalni podaci. Mogućnost delimičnog oporavka oštećene arhive je prilično „tugaljivo" pitanje – uspeh zavisi od toga gde je oštećenje nastalo. ARJ arhive se, naime, sastoji od glavnog zaglavlja (50-tak bajta) i niza „slogova" koji odgovaraju pojedinim arhiviranim fajlovima. Svaki slog se sastoji od lokalnog zaglavlja i samih arhiviranih podataka. Ako se oštete podaci odgovarajući fajl se ne može dobro raspakovati i tu nema mnogo pomoći – fajl je neispravan, ako je u njemu tekst vredi ga popravljati u editoru, a ako je u pitanju EXE program, obično možete samo da ga „bacite". Prave nevolje nastaje ako se ošteti lokalno zaglavlje – podaci su tada u suštini ispravni ali se ne mogu uopšte raspakovati jer nedostaju neke statusne informacije. Da bi stvari bile još gore, na taj način je poremećen „lanac" lokalnih zaglavlja tako da ARJ ne može da raspakuje ne samo oštećeni fajl, nego ni fajlove „iza" njega koji su u stvari sasvim ispravni!

Jung se u verziji 2.41 potrudio da reši neke od ovih problema. Do sada je naime, postojale opcije **-jr** koje je raspakivala datoteke sa CRC greškama (takvi fajlovi se normalno brišu) ali sada postoji i **-jr1** koja oporavlja i arhive sa oštećenim lokalnim zaglavljima. Delovanje ove opcije nije nerocito dokumentovano tako da nam je ostalo samo da probamo „ručno" smo (pritnenom disk editora iz paketa *Norton Utilities*) oštetili nekoliko lokalnih zaglavlja i pokušali oporavak. Jedenput smo uspeli dva puta nismo uspeli da oporavimo tekuću daloteku ali jesmo one iza nje, dok je u četiri slučaja i ostatak arhive bio totalno neupotrebljiv. Sve u svemu, videćemo kako će se ova opcija ponašati kod stvarno (a ne veštački) oštećenih arhiva – rekli bi se da je učinjen korak u pravom smeru, ali da stvari ipak nisu doterane do kraja. PKZIP je, kada se radi o oporavku oštećenih arhive, u boljoj poziciji jer njegov autor, uz sve lokalne hedere na kraj arhiva dodaje i centralni direktorijum sa podacima o svim kompnmovanim datotekama. Time je rezultujuća arhiva produžena (što kompenzuju superiorni algoritmi kompresije koje PKZIP podržava) i povremeno se desi da arhivirana datoteke može još jednom da se arhivira za neki promil (u praksi nebitno, ali deluje pomalo neozbiljno) ali se dosta i dobilo pre svega brzine pregleda arhive a zatim i veća mogućnost oporavke – čim je neki važan podatak ponovljen na dva mesta (u lokalnom zaglavlju i centralnom direktorijumu) sigurnost je povećana! Na žalost ni PKZIP ne konsti ovu blagodet tj. izgleda da PKZIPFIX „ne ume" da oporavi lokalno zaglavlje na osnovu centralnog direktorijuma! Moguće je da se registrovanim vlasnicima PKZIP-a stavija na raspolaganje još neki „doktor za .ZIP-ove" (tako je bilo kod verzije 1.10) kada se radi o *shareware* paketu. PKZIP i ARJ su sasvim slični po (ne)mogućnostima oporavka oštećenih arhiva. Ukratko kada arhivirate na diskete, koristite opciju **ARJ -jt** i pri samom upisu testirajte ispravnost obavljenog posla – jeste da ćeta izgubiti koji minut, ali ćete se osećati mnogo sigurnije!

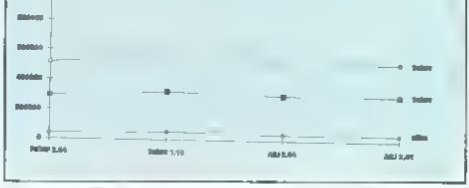
BESPLATNO... ZA POČETAK

ARJ se kao i većine drugih arhivera za PC računare distribuira kao *shareware* – možete ga slobodno preuzeti sa svih vecih BBS-ova a zatim (uvek u originalnom obliku) distribuirati prijateljima sistu na BBS ove i uključivati u *public domain/shareware* biblioteke. Autor je stavio javnosti na raspolaganje i UNARJ.EXE koji

Slika 2

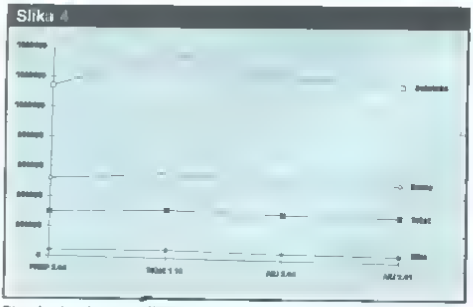
Solidne prednosti. PKZIP primetno brže arhivire i pri standardnoj kompresiji

Slika 3



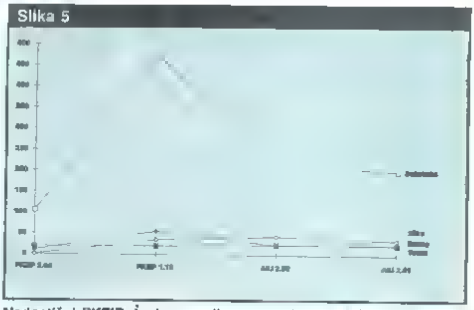
Ujednačen rezultat: ne koliko se kilobajta datoteka može svesti maksimalnom kompresijom

Slika 4



Standardna kompresija: koliko se datoteka može koprimoveti primanom standardnog stepena kompresije

Slika 5



Nedostižni PKZIP: kada se radi o vremenime makaimalne kompresije, PKZIP 2.04g je neprosto nedostižan!

omogućave raspakivanje arhiva kreiranih pomoću ARJ a – ovaj program možete potpuno slobodno distribuirati bez ikakve licence U „paketu" je i *C source* ovoga programa, na osnovu kojeg se ARJ format može podržati u drugim programima

Ukoliko ozbiljno koristite program više od 30 dana, treba da počnete da razmišljate o registraciji koja se svodi na uplatu sume od 40 dolara autoru, na adresu *Robert Jung 2606 Village Road West, Norwood, Massachusetts 02062 USA* (obaveštenja možete dobiti i elektronskom poštom, na adresu *robjung@world.std.com* ili *72077.445@compuserve.com*) Poslovna primena (koja obuhvata i distribuciju sopstvenih komercijalnih programa u vidu ARJ arhiva) kao i primene u poslovnim i državnim organizacijama zahtava registraciju koja se posebno dogovara sa autorom u zavisnosti od toga na koliko će se računara ARJ koristiti i u koja svrhe

Među pogodnostima koje pripadaju registrovanim korisnicima ARJ-a spadaju i dodatni uslužni programi kao što su ARJR (ARJ bez HELP-a i mogućnosti kreiranja arhiva koje se same raspakuju – zgodan zbog toga što troši manje memorije od „punog" ARJ-a pa sa može pozivati iz sopstvenih programa) i DEARJ (deje „skresani" ARJR koji samo raspakuje datoteka) Glavna stvar je međutim pravo na kupovinu takozvanih „zaštitnih omotača" (*security envelope*) koji se naplaćuju 50 dolara po komadu – morate dakle da platite 40 dolara za registraciju **plus** 50 dolara za „omotač" Šta je zaštitni omotač? Ako u njega „umotate" programe ili podatke koje ćete dalje distribuirati, recimo preko BBS-ova, svako raspakivanje arhive će proizvesti poruku da je podatke "zapakovao" taj-i-taj pojadinac ili te-i-te firma. Ako neko u međuvremenu promeni makar jedan bajt arhive ARJ će to primetiti i izdati odgovarajući izveštaj, tako da će eventualni korisnik biti upozoren da program koji je preuzeo svakeko nije „verodostojan" To je naročito značajno za antivirus programe koji su, kako se pokazalo često „napadani" od hakera i pretvarani u opasna trojance ili čak druga viruse – glavnina posla u borbi protiv virusa je de budste sigurni da slatke koje konstite nisu još opasnije zaražene!

Nevolja se ovim sigumosnim omotačima je što često pružaju lažnu sigurnost – mnogi haken relativno brzo „razbiju" sigurnosni mahanizam i kreiraju program koji omogućava bilo kome da registruje arhiver pod svojim ili što je mnogo opasnije pod tuđim imenom Takav program se onda postepeno širi po piratskim BBS-ovima kojih nije malo ni u Sjedinjenim Državama ni u drugim zemljamo Imeli smo prilke da se uverimo da su haken „provalili" sigumosnu šemu svih verzija PKZIP-a (pa i ova poslednje 2.04g) kao i, što sam Jung priznaja ranijih verzija ARJ-a Zato je u verziji 2.41 *Paul Kocher* dizajnirao potpuno novi mehanizam koji će, aventualno duža odoleveti napadima Iako je njegov format po prvi put javno dokumentovan Međutim, u dokumentaciji se redovno pominja da zaštitni omotači nuda „umerenu" zaštitu arhiv što bi moglo da znači da je i sam autor svestan da je nemoguće obezbijediti zaštitu njen sadržaja u duhu „nost će pokazati koliko je ta sumnja opravdana

Specijalitet ARJ-a je da nece ni raspakivati daloteku čiji je „zaštitni omotač" oštećen ZIP prijavi problem ali normalno raspakuja podatke Ako baš želite da raspakujete arhivu koju su „poseli" hakerski prsti konstite opciju **-he** i budite oprezni sa sadržajem! Ukoliko je arhive kreirana nekom od ranijih varzija ARJ-a, ova mera nije potrebna. pošlo ARJ 2.41 ignoriše eventuelni zaštitni omotač kreiran nekom od ranijih verzija koje su već proglašena za „nesigurne"

ZIP ili ARJ

Završiti ovaj prikaz komplimentima na račun ARJ-a ne bi bilo ni malo leško Čitaoce će međutim mnogo više zanimati mesto ARJ-a na tržištu tj njegovo poređenje sa konkurentskim programime sličnog tipa naročito sa izuzetno popularnim PKZIP-om 2.04g To poređenje nije ni malo jednostavne stvar – oba arhivera su veoma dobra i može se preporučiti upotrebe bilo kog od njih Ipak daćemo nekoliko generalnih napomena o raznim aspektima njihovog rada

Što se tiče osnovnog parametra, procenta arhiviranje, mišljenja smo da je on ujednačen – na našim testovima PKZIP je postigao nešto bolje rezultate, ali su razlika mišlavna ako se izraze u procentime dužine originalne datoteke PKZIP i ARJ, u celini gledano proizvoda arhive približnih dužina

PKZIP je u osetnoj prednosti po pitanju brzina arhiviranja – koristeći kompletnu memoriju računara (pa i EMS/XMS), autor PKZIP-a je uspeo da buda brži za 20, 30 i više procenata nismo uspeli da pronađemo ni jednu datoteku koju bi ARJ arhivirao brže od PKZIP-a bez obzira na primenjeni stepen kompresije

ARJ „dolazi na svoje" kada je reč o fleksibilnosti – brojne komande i opcije doduše uz neminovnu cenu složenije upotrebe omogućavaju da uz pomoć ARJ a uradite gotovo sve što vam padne na pamet Prednosti se naročito uočavaju kod rada sa višedelnim arhivama, gde PKZIP daluje sasvim „amaterski" u poređanju sa ARJ-om Iako razumemo težnju da se arhiviranja učini jednostavnim i da se izbegnu brojne komplikovane opcije nije nam jasno zašlo autor PKZIP-a nije obezbedio makar najosnovnlju mogućnost kreiranja arhiva proizvoljne dužina na disku ARJ ostaje nezamenljiva alatke za to

Kada se radi o sigurnosti skloni smo da oba arhivera proglasimo visoko pouzdanim I jedan i drugi su proživeli "cme dane" u kojima su nastajale neispravne arhive (ARJ u verziji 2.20 a PKZIP u verziji 2.04c) ali su sada sve te „dečje bolesti" uglavnom stvar prošlosti nisu primećeni problami sa pouzdanošću Pri oporavku oštećenih arhiva, PKZIP je u blagoj prednosti zbog ponavljanja podataka iz lokalnih zaglavlja u centralnom katalogu ali priložene *sharewara* alatka PKZIPFIX na „ume" da iskonsti te duplirane podatka, tako da se ne oseća neka neročita prednost nad ARJ-om koji te podatke i nema

A rasprostranjenost? PKZIP je tu svakako, u velikoj prednosti – njegov autor imao je sreće (i veštine) da se nađa u pravo vreme na pravom mestu se pravim proizvodom dok za ARJ mnogi ni danas nisu čuli pa ga čak ni neki populami programi (npr *PC Tools*) ne podržavaju Ipak ARJ je zacrtao odrađene trendove koji drugi sada sustižu ili još neko vreme neće sustići – zalo je njegova popularnosl u stalnom porastu Slična logika verovatno može da se primani i na domaće uslove ali samo kada se radi o kompjuterame koji se koriste u srednjema u kojime se na zna mnogo o arhivanma. Medu programenme, ARJ je izuzetno popularan, možda i popularniji od PKZIP-a. Na Sezamu smo, na primer nedavno organizovali glasanje za „najbolji" arhiver i PKZIP je pobedio sa svega nekoliko glasove razlike!

Naš zaključak bi, sva u svemu išao blago u korist PKZIP-u ceim kada se radi o kreiranju arhiva iz više delova U svakom slučaju, i PKZIP i ARJ nalazićе se u vašem SYS ili UTIL direktorijumu oba su potrebna, jer će vam često sh'zati arhive oba tipa. Koji od njih ćete konstiti za arhiviranje sopstvenih datoteka zavisi od mnogih stvari a naročito reki bismo od vašeg ukusa A o ukusima zna se ne vredi mpogo raspravljak

PUTOVANJE KROZ LJUDSKO TELO

Kao što ja svima poznato, računari su našli primenu u skoro svakoj ljudskoj delatnosti. Ne treba pominjati koju pomoć ljudi imaju u računarima u oblasti crtanja, pisanja, komponovanja, projektovanja štampanih ploča... A da li će jednog dsna računari potpuno (uz pomoč svojih produženih ruku – robota) zameniti čoveka? Pa, kako stvari stoje, verovatno hoća. Onda će čovek imati dovoljno vremena da ratuja, pardon, uživa u dokolici, okružen lepoticama koje ga hrana egzotičnim voćem... Pljus! Dečko, probudi se... opet sanjas.

Jurij Titav

Većine (neukog) stanovništva je uvek u lekaru doktoru, uglavnom osobi koja je imala dovoljno petlje da čačka po ljudskom telu raznim skalpelima, zaliva ga raznim tečnostima, tera ga da guta raznobojne pilule – videle Bogal Naravno i ta dragocena ljudska veština je u stvari samo dobro utrenirana rutina – i ništa više. Nešto slično Pavlovljevom refleksu doktore, boli me ovde i ovde, a ovde me probade, Aha, to je to...i to. Sve se uglavnom svodi na određeni skup simptoma koji se presikavaju u opet, određeni skup dijagnoza. Pa eko pacijent preživi – živela medicina, a ako ne onda je kriva sudbina ili baba Drage koja je bacala kletve.

Danas se na tržištu nalazi obilje softvera koji će vam postaviti adekvatnu dijagnozu na osnovu ukucanih simptoma. I da li je takav program bolji ili gori od prosečnog lekara? Pa zavisi kome više verujete. Međutim da se ne bismo ovde bavili raznim pretpostavkama najbolje je to naše telo upoznati što bolje (muški deo čitalaca poznaje onu lepu kneticu iz vojske prilikom podele prakomandi. Upoznaj svoju domovinu da bi je više voleo, Trpče Nikolovski – Pivka) jer hteli mi to ili ne, postojaćemo na ovoj planeti dokle god naše telo bude funkcionisalo. A one legende tipa „ON će večno živeti u nama – " ipak treba batiti...

Predamnom se našao vrlo zanimljiv program, *BodyWorks v2.1 for Dos*. Dolazi na 2 HD diskete 5.25", instalacija traje par minuta i posle nje ostajete uskraćeni za nekih 2.3Mb prostora na vašem hard disku.

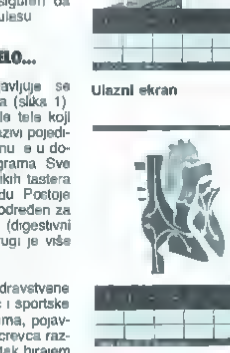
Program radi u grafičkom režimu i preporučena je color VGA grafika. Nažalost i pored preporuke, ekran je stalno u režimu sa 16 boja koje ipak ne umanjuju kvalitet slika. Ne znam da li je podržana hercules grafika (a i nisam se mnogo interesovao), ali sam siguran da program ninašta ne bi ličio na herkulesu.

PROŠETAJTE KROZ SVOJE TELO...

Po aktiviranju programa pojavljuje se ekran (uslovno) podeljen na tri dela (slika 1). U levom delu se nalazi prikaz dela tela koji smo izabrali, desno su tekstualni nazivi pojedinih delova tela koji se vide na ekranu a u donjem delu ekrana su komande programa. Sve komande su prikazane pomoću velikih tastera sa slićicom koje asocira na komandu. Postoje dva reda takvih „dugmića". Prvi je određen za direktan pristup raznim sistemima (digestivni sistem, kardiovaskularni itd.) a drugi je više funkcionalniji.

- Prvim tasterom pozivamo četiri zdravstvene teme – AIDS, droge, prvu pomoć i sportske povrede. Biranjem neka od tih tema, pojavljuje se sledeći meni koji u sitna crevca razleže izabranu temu u podteme i tak birajem podteme dobijemo ekran prepun teksta koji detaljno obrađuje izabranu temu. De dam mali primer pod drogama su se svrstali i kofein i alkohol i nikotin, a provlače se i marihuana i coca-cola, pardon kokain. Jedini zaključak koji iz ovog spiska izvlačim je da su pomenute droge najrašireniie i da uzrokuju najveće probleme među radnim ljudima i građanima. Na žalost, ovde nema slika osoba koje su dugo uživale određenu drogu, e baš bi bile vrlo poučne.
- Sledeći taster nas dovodi do tri animacije kako rade pluća, kako radi srce (slika 2) i kako radi mišić (ne Onaj mišić). Animacije su jednostavne ali se bar stiće osnovni pojam „kako sve to radi"
- Zatim dolazimo do testera koji nas „vraće korak unazad" ako odemo suviše „duboko" u telo (Back step)
- Dva naredne testera nam omogućavaju da zumiramo sliku pojedinog organa ili dela tela (tj. da je razvučemo preko calog ekrana – Full View) i netenane razgledamo (slika 3). Možemo da upotrebimo i neki screen-grabar da bismo je „skinuli" sa ekrana i kasnije koristili. Naravno, autori programa su i to predvideli ali malo nevešto jer sa drugim testerom jeste da možemo da snimimo sliku u PCX formatu na disk ali snima se i slika dela tela i tekstualni deo koji stoji pored slika (znači levi i desni deo ekrana, bez tastera). Pa, možda će i to nekom trebati.

Ulazni ekran



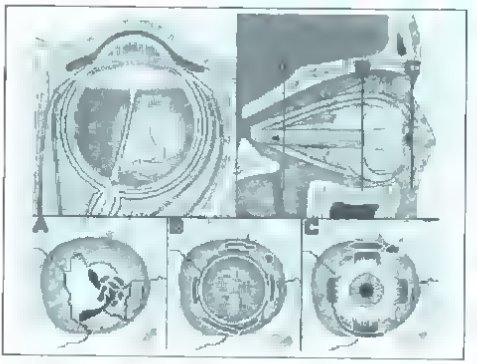
Animacija rada srca

- Pored „skidenja" slike u detoteku moguće je i njeno štampanje što nam i omogućeve sledeći taster. Pre štampe je moguće podesiti koji štampeč (ili kompatibilac) posedujemo. Ponuđeni su 9 i 24 pinski Epson-i i HP LJ II i III. Verovatno se smetre de svi ostali štampeči prosto moraju biti kompatibilni sa ponuđenim, inače se i ne smatraju štampačima. Otisak na 24-o pincu i nije još mada je više informativan. Valjda sam se raznazio otiskom *CorelDrew*-a ili neke druge *Windows* aplikacije. Vrlo je neobično da ovakav program, sa vrlo atraktivnim i detaljnim crtežima nema podršku za neki štampač u boji?! Jeste da se u okviru help-a pominje 1991. godine za koju se opet ništa ne kaže (da li je tada nastala prva verzija programa ili ova, 2.1), ali ja sam na poslu još davne 1988. godine imao HP PaintJet štampač („pljuckajući" u boji) što znači da su uz malo truda mogli da omoguće direktnu izradu atraktivnih slika na papiru. Naravno, rešenje uvek postoji, snimiš sliku u formatu PCX, učitaš je u neki „crtači" program i tek onda odštampaš na odskvetnom štampaču. Ali to ipak nije To.
- Sledeći taster bih nazvao „pogledaj još i ", tj. bez obzira u „kom delu tele" se nalazimo, ovim tasterom uvek postižemo da nam se ponudi spisak srodnih funkcija/delova tela koji su na neki način u vezi sa ovim što upravo gledamo/proučavamo. S obzirom da teko možemo stvarno da se zapetljamo, tu nam dobro dolazi pomenuti taster „vrati korak unazad".
- Naredni taster je možda najkorisniji jer nam omogućava da pozovemo dijalog za traženje pojedinih delova tela na osnovu jedne (ili više) reči. Znači, standardna *search* mogućnost. Rezultat traženja je spisak svih tema gde se zadata reč pominje. Naredna akcija bi bila biranje željene teme i kasnije razgledanje pojedinog organa ili čitanje o njemu.

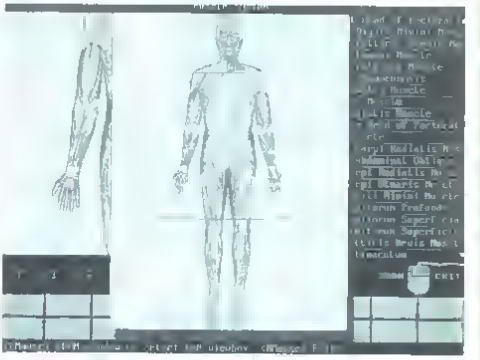
Poslednja dva tastera su znak pitanja tj. standardni help i Exit.

...DA BISTE GA BOLJE UPOZNALI

I najzad reč dve o samom radu sa programom i šta može da nam pruži. Naravno u ovoj fazi program ne može biti neke velike pomoć majstorima koji popravljaju ljude. Ciljna grupa bi bila učenici medicinskih škola, studenti početnih godina medicine i naravno masa PC korisnika koji nemaju pojma šta se u njihovom telu nalazi i kako to telo radi, a imaju žarku želju da to saznaju.



Zumiranje do najsitnijih detelja



Prikaz celog tela

Po aktiviranju programa nalazimo se odmah u nervnom sistemu. Na ekranu se nalazi glava sa ramenima. Pristup ostalim delovima tela je jednostavan: ispod slike se nalaze kursorski tasteri kojima se šetamo po telu u sva četiri pravca. Takođe se tu nalazi i taster FULLBODY koji nam omogućava da prikažemo celo telo i onda mišem obeležimo deo koji želimo da vidimo tj. detaljnije pogledamo (slika 4). Interesantno je da ovaj program insistira na posedovanju miša jer se svaka operacija obavlja sa njim. Moguće je koristiti i tastaturu, ali se to svodi na par *hot key* parova (na pr. *Alt-F –>> fullbody, Alt-A –>> animation help* itd.)

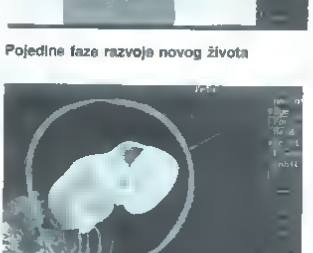
Mišem se krećemo po celom ekranu, u donjem delu, ukoliko nisu „pokvareni" tekstom prisupamo komandnim tasterima u desnom delu gde je ispisan tekst, jedan klik mišem uspostavlja vezu (u obliku prave linije) između teksta i dela tela na koji se tekst odnosi. Uspostavljenjem veze, na crtežu miša koji se nalazi u desnom delu ekrana ispod teksta, pojavljuju se oznake i to ako se pritisne levi taster miša, dobijamo sliku a pritiskom na desni taster dobijamo kratak opis tog dela tela. Naravno, neće uvek biti dostupna slika i/ili tekst jer sve zavisi koliko su autori išli u „dubinu" tele. Vizuelne osobe koje ne vole da čitaju, mogu mišem da se pozicioniraju na samu sliku dela tela i odmah po „ulasku" u zonu slike na istoj će se pojaviti na pojedinim delovima tele znak u obliku, pa de ga nazovem, romboidnog mišanja. Dovlačenjem miša na taj znak i pritiskom na levi taster, uspostavlja se prevolinijska veze između slike i teksta, a sledeći klik već zavisi da li želimo sledeću sliku (eko postoji) ili kratak tekst.

Same slike su i ako u svega 16 boje vrlo atraktivna i detaljne i stičam utisak de su ih skenirali iz nekog anatomskog (papirnog) atlasa. Ovde odmah i male kritika: program bi bio mnogo atraktivniji kada bi postojao pravi „zoom u telo" gde bi konsnik mogao da uvećiva sliku maltene do same ćelije. Ali onda program verovatno na bi imao 2Mb nego celih 20Mb. Takođe bi još zanimljivije mogućnosi bila kada bi postojale slike pravih organa tj. onakvi kakvi su oni u prirodi. E sad, koliko bi to bilo „ukusno" za nenavikntuteg posmatrača, ne znam. Jer, organi u prirodi se mnoogo razlikuju od ovih divnih i nevinih slićica. Možde većini ne bi izdržao stomak kada bi videli neku sluzavu, bezličnu masu za koju neko tvrdi de je to jetra? Bolje živeti u blaženom neznanju. Bebe donose roda, zar ne? Uostalom, koliko vas, srećno ožanjenih muževa tranžira tek kupljeno krvavo meso? Sve prepuštate „slabijem" polu kao da se ne isprljate, a tu je i sveža „Politika" koju treba prelistati.

Pristup pojedinim sistemima u ljudskom telu se obavlja putem prvog reda tastera u donjem delu ekrana. Autori su celo funkcionisanje čoveka podelili na 10 sistema: skeletni (koske) digestivni (ili sistem za ishranu), mišići, limfni sistem, endokrini (žlezde se unutrašnjim lučenjem), nervi, kardiovaskularni sistem, urinarni sistem, muški reproduktivni sistem i ženski reproduktivni sistem. Vrlo korisno je što kada izaberete da vam se prikaže na primer deo tela od vrata do pupka, te silueta ostaje stalno na ekranu, a vi možete birianjem određenih „sisteme" da gledate šta se u tom predelu nalazi. Pojedini sistemi imaju svoju „prednju" i „zadnju" stranu, kao na primer skeletni sistem ili mišićni sistem radi lakšeg i preciznijeg pogleda, e i detaljnijeg objašnjenja.

Pored samog „atlasa" ljudskog tela i opisa funkcije svih organa date su i pojedine faze razvoja novog života (slika 5). Zanimljivo je da do takvih stvari nije moguće doći preko raznim menija ili tastera nego korisnik mora da se „pomuči" i potraži ih preko *Search* funkcije. Predlažem da potražite reč *fetus* i od ponuđenih tema izaberete na primer eye. Pred vama će se pojaviti slike unutrašnjosti majčinog tela i fetusa od par meseci (slika 6). Slika je vrlo upečatljiva i strašno mi je žao što nije podržano više boja.

Pojedine faze razvoja novog života



Fetus od nekoliko meseci

JOŠ DA JE POD WINDOWS-ima...

Na kraju de dam globalnu kritiku programa. Vrlo lepo urađeno sa puno atraktivnih slika, stručnih objašnjenja (na engleskom jeziku naravno). Povremeno se pojavljuju i latinski nazivi tamo gde je potrebno ali uglavnom preovlađuje razumljivi engleski jezik. Naravno potrebno je znati kako se koji organ zove na engleskom/latinskom što već može biti problem jer mnogi ne znaju ni kako se to zove na srpskom. Preporučujem program svim radoznalim ili onima koji se stide da prijau stanje ili prosto onima koji žele da nauče nešto više o svome telu a „prinuđeni" su da sede pred monitorom ceo dan. Pristup podacima/slikama je brz, program nije pravio nikakve probleme (u smislu blokiranja mašina, samoresetovanja, *devide by zero* i slično). Sve u svemu jednostavno i zanimljivo.

Negativne kritike bi bile samo u smislu poboljšavanja postojeće koncepcije: odnosno samo prelazak na SVGA standard (bar 256 boja, rezolucije može i da ostane 640x480), dodavanje dinamičkog *zoom*-a, dodavanje digitalizovanih slika delova tela i organa. I naravno prebacivanje na *Windows* platformu, jer bi tek onda *BodyWorks* postao ono pravo. Ala bi se štancovali megistarski i doktorati... Mnogo više nego do sada.

ADA computers

BEOGRAD
Tadeuša Košćuškog 72
tel/fax: 011/186-267; tel:011/186-355

NOVI SAD
Siriška 42
tel/fax: 021/416-189

286 20 MHz, 1MB	990
386SX 33 MHz, 2 MB	1.100
386 40 MHz, 4 MB 64K CACHE	1 320
486 33 MHz, 4 MB 256K CACHE LB	1.990
486 50 MHz, 4 MB, 256K CACHE	2 300
486 50 MHz 4 MB 256K CACHE VESA LB	3 470
486DX2 66 MHz 4 MB 256K CACHE VESA LB	3 770

Osnovne konfiguracije sadrže
40 Mb HDD, flopi disk 1,2 Mb, tastaturu, desk top kućište, monitor mono herkules 14" crno beli, 2S/1P port
modeli sa VESA LB *mini tower, SVGA kolor monitor sa VESA LB video karticom, HDD 210 MB*

DOPLATE

HARD DISK 105 Mb	150
HARD DISK 120 Mb	190
HARD DISK 210 Mb	400
VGA MONO MONITOR 14" SA VGA KARTICOM	80
KOLOR MONITOR SVGA 1024 X 768 14" 512 kb	360
KOLOR MONITOR SVGA 1024 X 768 14" 1 Mb	430
KUĆIŠTE MINI TOWER	20

NOTEBOOK

386SX/25/2MB /80 MB HDD/FDD 1 44/MONO VGA	2850

OSTALA OPREMA

EPSON LX-400/800	420
EPSON FX-1170	1250
LASERSKI ŠTAMPAČ HP IV	3500
KABL ZA ŠTAMPAČ	30
YU SET ZA LX MODELE	30
KUĆIŠTE DESK TOP	180
KUĆIŠTE MINI TOWER	200
KUĆIŠTE MIDI TOWER	330
KUĆIŠTE TOWER	400
MATIČNA PLOČA 286-20	100
MATIČNA PLOČA 386SX-33	210
MATIČNA PLOČA 386 40 CACHE 64K	310
MATIČNA PLOČA 486-33 CACHE 256K	980
MATIČNA PLOČA 486 50 CACHE 256K	1290
MATIČNA PLOČA 486-50 CACHE 256K VESA LB	1380
MATIČNA PLOČA 486-66 CACHE 256K VESA LB	1680
I/O KARTICA 1PAR/2SER	30
MGP (Herkules) KARTICA	30
VGA 16 BITNA 512K KARTICA	90
VGA 16 BITNA 1Mb KARTICA	160
VGA LB TSENG LABS 1MB	230
VESA LB 1MB KARTICA	250
AT IDE KONTROLER	30
KOMBI KONTROLER	60
1Mb RAM MODUL	90
256K RAM MODUL	30
TASTATURA KLIK 101	80
FLOPI DISK 1,2 Mb 5,25"	140
FLOPI DISK 1,44 Mb 3,5 "	120
HARD DISK 40 Mb	250
HARD DISK 105 Mb	400
HARD DISK 120 Mb	440
HARD DISK 210 Mb	650
MONO MONITOR (Herkules)	330
VGA MONO MONITOR	350
KOLOR MONITOR SVGA 1024x768 14"	650
MIŠ SA PODNOŽJEM	40
ETHERNET KARTICA 16 BIT	190
STRIMER COLORADO JUMBO INTERNI 120 Mb	580
STRIMER COLORADO JUMBO INTERNI 250 Mb	670
STRIMER ARCHIVE INTERNI 150 Mb	1600
387-33 KOPROCESOR	150
MODEM INTERNI 2400	190
FAX/MODEM INTERNI 2400/9600	230
FAX/MODEM POCKET 2400/9600 MNP5/V42bis	500
UPS 500VA	980
TONER ZA HPIII	320
TRAKE ZA STRIMER	100
YU SET ZA HERKULES	30
FILTER ZA MONITOR MREŽNI	30

RADNO VREME OD 9 DO 17 ČASOVA, ISPORUKA ODMAH PO UPLATI
CENE SU U DEM, DINARSKA UPLATA PO DNEVNOM KURSU NA TRŽIŠTU U BEOGRADU
GARANCIJA 12 MESECI, SERVIS OBEZBEĐEN

DO POSLEDNJEG BAJTA

Sigurno ste navikli na standardna DOS formate disketa – 360 Kb i 1.2 Mb za veličine 5.25", odnosno 720 Kb i 1.44 Mb za 3.5". Da li ste znali da bez ikakvih problema možete dobiti diskete 20% do 130% većeg kapaciteta, uz ubrzanje rada sa njima od 30 do 100% i nesmanjenu pouzdanost? Obične 5.25" DD diskete umesto 360 Kb možete formatirati na 820 Kb, 5.25" HD na 1.48 Mb, a 3.5" HD na čak 1.72 Mbi

Jovan Bulajić

Sve ovo možete izvući bez gubitka pouzdanosti, na vašoj standardnoj konfiguraciji i najobičnijim disketama. Pristupak nije uopšte komplikovan potrebno je samo umesto DOS-ove komande FORMAT koristiti neke od programa koji diskete može formatirati na veći kapacitet. Sve što vam je potrebno možete za nekoliko minuta preuzeti sa Sezama, i potpuno besplatno koristiti.

Ovde nije reč o kakvom privilnom povećanju kapaciteta (u stilu *on-line* kompresije podataka) kao što to rade neki poznati alati za „povećanje kapaciteta" hard diska (*Stacker, Double Space*). Kapacitet se zaista fizički povećava, a mesta za to ima jer standardni formati koje je postavio PC/MS-DOS ne iskorišćavaju prostor na disketi do kraja.

Namera nam je da vas postepeno uvedemo u postupak formatiranja koji ćete nadamo se ubuduće redovno koristiti. Opisaćemo prvo strukturu diskete i njihovu podelu, proces formatiranja ograničenja DOS formata, da bismo vam na kraju pokazali kako bez po muke možete svoje diskete iskoristiti do maksimuma.

STRUKTURA DISKETA

Fizička struktura diskete je relativno jednostavna magnetni medijum (nosač podataka) smešten je u zaštitno plastično kućište na kojem se nalaze prorez (ili nešto složeniji mehanizam kod 3.5" disketa) kroz koji glava flopi disk jedinice (*floppy disk drive*) pristupa podacima koji su zapisani na disketi. Postoji nekoliko vrsta disketa zavisno od njihove veličine (broja strana i kvaliteta magnetnog medijuma

Podela disketa s obzirom na njihovu veličinu je najočiglednija. Veličina se izražava u inčima (oznaka „ 1 inch – 2.54 cm) a u svakodnevnoj upotrebi na PC računarima se koriste diskete veličine 5.25" (tzv „velika") i 3.5" („male")

Zavisno od toga da li podaci mogu da se pišu i čitaju na obe ili samo se jedna strana diskete razlikujemo dvostrane (*DS – double sided*) i jednostrane (*SS – single sided*) diskete. Ekvivalentno postoje dvostrane i jednostrane disk jedinice (detaljnije o strukturu flopi disk drajvova ovom prilikom nećemo pričati)

Diskete se delе i zavisno od gustine (*density*) zapisa tj. kvaliteta magnetnog sloja. Tako postoje diskete dvostruke gustine (*DD double density*) i veoke gustine (*HD high density*) HD imaju kvalitetniji magnetni sloj od DD i mogu podneti „gušće" zapisane podatke. Ove dva tipa se mogu lako optički razlikovati – rupa u sredini DD diskete je dodatno ojačana prstenom dok HD nemaju taj prsten

Ono što nas ovaj put najviše zanima je struktura zapisa na disketi. Podaci su upisani na koncentrično postavljenim krugovima koji se nazivaju traka ili staze (*tracks*) a svaka traka je podeljena na nekoliko sektora (*sectors*). Termin „traka" je ekvivalentan terminu „cilindar" koji se koristi za hard diskova

Trake i sektori se numerišu počevši od nule a flopi disk drajv zna poziciju nulteg sektora uz osnovu malog proreza (*index hole*) pojed rupe u sredini diskete

Maksimalan broj traka i sektora koji neka disketa može fizički podneti zavisi od njene gustine (da li je HD ili DD) a broj traka i sektora koji se koriste zavise od načina na koji je disketa formatirana.

FORMATIRANJE

Formatiranje je neophodan postupak inicijalizacije koji treba izvesti da bi neka nova disketa mogla da se koristi Pri formatiranju se proverava ispravnost disketa i upisuju neki podaci od značaja za operativni sistem (npr u MS-DOS-u to su *boot* sektor osnovni direktorijum tabela alokacije datoteka – FAT)

Formatiranje se obično obavlja posebnim programom u MS-DOS-u za te je zadužena komanda FORMAT Formatiranje se može obaviti i iz nekih uslužnih program (*PC Tools*) a u nekim BIOS-ima u setapu ili delu za dijagnostiku postoji funkcija za te (mi smo ranije imali jedan takav slan AMI BIOS)

Prilikom provere ispravnosti diskete može se desiti na program za formatiranje ne može da pročita ili upiše podatke u određeni sektor – lebe se taj deo diskete označava (markira) kao loš sektor (*bad sector*) i operativni sistem ga zaobilazi prilikom kasnijeg upisivanja podataka.

Za smeštanje svojih podataka operativni sistem (ili program za formatiranje) koristi nulu (0) traku Ukoliko je kojim slučajem ta nulte traku loša DOS neće moći na nju da upiše vitalne podatke pa se lakva diskete na može lor mahrati uobičajen način U tom slučaju će vas DOS pozdraviti porukom *Invalid media or Track 0 bad disk unusable*

Ukolike pokušate da priskupite nekoj disketi a DOS vam javlja grešku „General failure" onda najverovatnije tu disketu niste ni formatirali

Proces formatiranja i logička organizacija disketa su različiti ze razna operaciona sistema Zbog toga se dešava sla identične diskete (npr 3 5" DS/HD) formatirana čak na isti kapacitet (npr 1 44 Mb) ala na različitim računarima (PC i Mac) ue mogu da se čitaju na računaru drugog tipa. Ukoliko mate problema se prebacivanjem sadržaja sa jednog tipa računara na drugi pogledajte alatke koje se nalaze na Sezamu u direktorijumu DISKUTIL tamo možete naći programe koji olakšavaju rad se Macintosh CP/M i drugim disketama na PC u

DOS FORMATI DISKETA

Kada je davake 1981 godine IBM predstavio prvi PC računar i PC-DOS (kasnije i MS DOS) verziju 1 0 predstavljen su i prvi formati diskete koje je on koristio Bile su to jednostrane diskete veličine 5 25" (u to vreme su još uvek vrlo tadašnje bile ogromne diskete od 8 inča tj 20 cm) formatirane na 180 Kb.

Nešto kasnije PC-DOS verzija 1 1 donele je podršku dvostranim disketama od 320 Kb a pojava PC/XT računara (1983 PC-DOS 2.0) jednostranom formatu od 180 Kb i dvostranom od 360 Kb Ovaj posledujl – 360 Kb na DS/DD disketi je dugo bio apsolutni standard za PC računare

Pojavom PC/AT računara I kasnije PS/2 serije IBM je uveo dva nova standardna formata disketa koji su danas u apsolutnoj dominaciji na tržištu To su 5 25" diskete kapaciteta 1 2 Mb i 3.5" kapaciteta 1 44 Mb Obe ova formata odnose se na dvostrane diskete veoke gustine (DS/HD)

Poslednji novitet su 3.5" diskete kapaciteta čak 2.88 Mb ali ovaj standard još nije zaživeo pre svega zbog visoke cene disketa i dodatnih hardverskih zahteva (uz disketnu jedinicu potrebni su i disk-kontroler BIOS novoe generacije koji podržavaju ovaj format) Očekujemo da razvojem tehnologije u sledećih nekoliko godina format od 2 88 Mb postane standard za PC računare ake se do tada ne pobolša tehnologija kompakt diskova (CD)

U Tabeli 1 su prikazani osnovni podaci o svim dosadašnjim formatima disketa koje podržava PC/MS-DOS

Pošto smo se već upoznali sa podelom diskete na sektore i trake i sa osnovnim DOS formatima disketa rad je da se upoznamo i sa načinom na koji se stvarano vrši formatiranje kapacitet neke diskete

Ukupan broj sektora jednak je proizvodu broja strana, traka I sektora po traci Ú slučaju dvostrane 5.25" diskete formatirane ue 360 Kb se 40 traka I 9 sektora po traci ukupan broj sektora jednak ja 2 × 40 × 9 što iznosi 720

Standardno se na svaki sektor zapisuje 512 bajta podataka što znači da se na 720 sektorni može zapisati 720 × 512 bajta tj ukupno 368 640 bajta ili 360 Kb Kako je 1 Kb = 1024 bytes 2 × 512 bytes znači da dva sektora imaju ukupnu dužinu od jedan kilobajt iz čega sledi da je kapacitet diskete izražen u kilobajtima jednak polovini broja sektora

Dakle konačna formula (1) za dobijanje ukupnog (formatiranog) kapaciteta diskete izraženog u kilobajtima glasi

kapacitet = (broj strana × broj traka × broj sektora po traci) / 2 [Kb] (1)

Ukoliko je reč o dvostranim disketama (što je golovo isključivo slučaj) skraćivanjem broja strana (koji je u tom slučaju 2) u brojiocu i dvojke u imenocu dobijamo još prostiju formulu (2) za izračunavanje kapaciteta *dvostrane diskete*

kapacitet = broj traka × broj sektora po traci [Kb] (2)

$00 II

Verzija	1 80 14 07 91
Autori	Alberto Pasquale
Sezam	
katalog	r:\bmpc\diskutil\
arhiva	800ii180.zip
dužina	14.133

FDFORMAT

Verzija	1 9 21.07 91
Autori	Christoph H. Hochstötter
Sezam	
katalog	r:\bmpc\diskutil\
arhiva	fdform19.zip
dužina	109 056

FMT

Verzija	1 0 13 03 92
Autor	Oleg Klimov
Sezam	
katalog	r:\bmpc\diskutil\
arhiva	fmi100.zip
dužina	35 070

BOLJE OD DOS-a

Formatirani kapacitet disketa je manji od onog što magnetni medijum može da izdrži pošto se na kraticle cela dostupna površina u maksimalne gustine zapisa Ćeste se na kutijama disketa sreću natpisi kao što je „2 Mb kapacitet" ali to je *neformatirani kapacitet* s predstavlja teoretske maksimum Dri se ne može izvući ali se ipak može iskonsti znatno više nego što to MS-DOS dopušta.

Prvi pokušaj za bolje iskorišćenje diskete bili su formatiranje DD disketa (od sada ćemo podrazumevati da je reč o dvostranim tj DS) u HD disk drajvovima Dakle pokušavano je de se DD diskete koje DOS formatira na 360 Kb formatiraju na 1 2 Mb Činjenica je da je kvalitet magnetnog sloja ue DD diskelama slabiji nego na HD i da se na meže očekivati potpun uspeh ali dobici su ipak bili solidni.

Ovakvim postupkom (koji je ranije i kad nas bio prilično rašireni) uspevala su se dobri diskete se ake 500 Kb pa čak i do 1 Mb korisnog prostora dok je ostalak prostora bio popunjen lošim sektorima. Ovi loši sektori (*bad sectors*) bili su koncentrisani bliža sredini diskete jer je tamo zapis najgušći (prečnik kružnih traka sa prema sredini smanjuje dok broj sektora po traci ostaje konstantan) Pri ovakvom formatiranju se mogla čuti karakterističena buka (tzv „dranje zastа (e laiko i zvučalo) pri nalaženju glave disketene jedinice ula loše sektore

Pouzdanost ovog rešenja bila je problematična take da smo se ovim trikom služili samo kad smo baš morali da sfisnemo mnogo podataka na jednu DD disketu

Onda su se pojavila programska rešenja koja omogućavaju pre svega veće iskorišćenje kapaciteta diskela, bez gubitka pouzdanosti rade Kod nas se uglavnom koriste tri programa te namene To su (hronološkim redosledom)

800 FDFormat i *FMT*. Osnovne podatke o njima možete naći u ličnoj karti uz ovaj tekst.

Osnovni način na koji se ostvaruju veći kapaciteti je jednostavan – omogućavaju formatiranje se povećanim brojem traka i sektora u odnosu na standardni. U novijim verzijama MS-DOS-a komanda FORMAT kao parametre prima broj traka *(/T:tracks)* i sektora po traci *(/N:sectors)*, ali ona može da "prođe" samo standardne vrednosti (npr. samo 40 ili 80 traka) inače će vas pozdraviti porukom *Parameters not supported by drive*.

TRI PROGRAMA

Od ova tri programa prvo su se pojavili program *800* odnosno njegov naslednik *800 II*. Namena programa *800* je kreiranje nekoliko novih formata flopi diskova uz očuvanje potpune DOS kompatibilnosti. *800 II* se instalira kao rezidentni program i zauzima manje od 1 Kb memorije (tačnije 976 bajtova). Radi tako što proširuje BIOS funkcije za rad se disketama, pa se većina proširenih formata može koristiti preko standardnih DOS komandi FORMAT DISKCOPY i DISKCOMP.

Kada je *800* instaliran DOS-ova komanda FORMAT će bez problema prihvatati i sasvim nestandardne vrednosti uz */T* i */N* parametra. Promena broja traka i sektora su i jedina opcija koje *800* pruža dok *FDFormat* i *FMT* imaju još dodatnih mogućnosti koje se mogu vrlo korisno primeniti. Zbog loga dosadašnjim kritikama programa *800* preporučujemo da pređu na *FDFormat* ili *FMT* kojima ćemo uglavnom i posvetiti ostatak ovog teksta.

Ukoliko ipak želite da instalirate *800 II* na Sezamu možete naći ova njegove verzije – 1.71 i 1.8. Verzije 1.8 je poslednja koje je izašla, ali ukoliko vam se desi da vam pravi problema (a dosta korisnika je prijavilo čudno ponašanje u Windows okruženju) uzmite nešto stariju verziju.

FDFormat je sledeći program koji se pojavio dok je *FMT* poslednji koji je zašao u uglavnom je pravljen po uzoru na *FDFormat*. Zbog toga je većina parametara za ova dva programa potpuno identična. *FDFormat* je posebno interesantan za programere jer je pružen kompletan izvorni kod u *Turbo Pascal-u* i asembleru.

Dodatne opcije koje *FDFormat* i *FMT* pružaju su uamerene na dodatno povećanje slobodnog prostora na račun sistemskog prostora koji odvaja DOS ubrzanje rada sa flopi-diskom poboljšanje i kod sektora. I ima i sasvim egzotičnih opcija kao što je povećanje razmaka *(gap)* između sektora. Nećemo spominjati sva opcije već samo one najkorisnije i potpuno bezbedne za korišćenje.

Za razliku od *800* kod ovih programa formatiranje se na vrši DOS-ovom komandom već posebnim programom – FDFORMAT.EXE ili FMT.EXE. Da te takva diskete bile čitljive potrebno je instalirati i mali rezidentni program koji proširuje funkcije INT 13 servisa (BIOS funkcije za rad se flopi-diskom). To su FDREAD.EXE ili INT13X.COM. Njihova instalacija će vam oduzeti najviše 200 bajtova memorije a u slučaju da koristite MS DOS 5.0 instaliran u visoku memoriju nećete izgubiti ni bajt pošto će se učitati na kraj DOS segmenta.

Posebno je zanimljivo da vam u većini slučajeva uopšte neće trebati taj rezidentni program! Ovo zavisi od mnogih faktora, pre svega od BIOS-a ali na našoj konfiguraciji (Phoenix 386 BIOS iz 89. MS-DOS 5.0) svi 5.25" formati rade bez ikakve potrebe za rezidentnim programima dok ih je za najviše 3.5" formate ipak potrebno učitati.

Ukoliko do sada niste koristili ni jedan od ova tri programa preporučujemo vam da odmah uzmete *FMT*. Najnoviji je Int13X je bolje urađen od FDRead a što je vrlo važno ima bolju verifikaciju ispravnosti diskete prilikom formatiranja. Ukoliko već koristite *FDFormat* i konzervativni ste neće biti nikakvih problema ni ako ostanete na njemu. Većina primera koje budemo navodili za svakodnevni rad odnosiće se baš na *FMT* njegovu sintaksu komandi.

VIŠE PROSTORA

Da se vratimo sada na osnovnu temu ovog članka – kako izvući najviše prošlora iz diskete? Na dve osnovna načina – povećanjem broja traka i povećanjem broja sektora po traci. Najveći dobici postižu se kombinacijom ove dve tehnike.

Glavnina dobitaka u prostoru postiže se povećanjem broja sektora po traci odnosno guščim zapljevanjem podataka. Ovo je moguće jer je magnetni medijum dovoljno kvalitetan (ima dovoljno magnetnih čestica po jedinici površine) da to podnese čak i bliže sredini diskete gde je i brzina zapis najgušći.

Za 5.25" DD diskete standardne je formatiranje na 9 sektora po traci ali se bez problema može iskoristiti 10 sektora dok se 5.25" HD umesto 15 mogu iskoristiti maksimalno na čak 18 sektora po traci. U 3.5" diskatama je magnetni medijum kvalitetniji (ili sam mehanizam diskatne jedinice precizniji) pa se na DD diskete umesto standardnih 9 bez problema može izvući 12 ili čak 13 sektora po traci a na HD do 21 umesto 18 sektora.

Ne bismo vam savetovali da pokušavate da izvučete još više traka po sektoru od ovih maksimalnih vrednosti koje smo naveli pošto i magnetni medijum ima neke granice preko kojih ne bi trebalo ići: pre svega zbog smanjene pouzdanosti. Ukoliko ipak pokušate da izvučete koji sektori više, očekujte da vam programi za formatiranje prijave gomilu loših sektora bliže sredini diskete tj. poslednjoj traci.

Što se traka tiče, razmak između koncentričnih traka je konstantan kao što je konstantna i njihova širina pa se povećanje broja traka na može ostvariti nekim „guščim" zapljevanjem. Maksimalan broj traka koje koriste standardni formati je 80, ali neke za proširenje ipak ima, zato što tima nije iskorišćena cela površina magnatnog medijuma ni prosečno ne može kretati glava disk jedinice. Diskete može da izdrži znatno više traka pa maksimalan broj koji se može iskoristiti zavisi od disk-drajva. U mehanizmu flopi-disk jedinice se obično nalazi graničnik koji ograničava prostor kojem glava (koje služi za čitanje i pisanje podataka) može pristupiti ali on je uglavnom tako postavljen da se bez problema može iskoristiti još nekoliko traka iznad osamdesete.

Na svim disk-jedinicama bi trebalo da je moguće bez problema koristiti do 82 traka. Može i više ne nekim drajverima čak i do 90 ali ova morate sami isprobati na svojoj konfiguraciji. Pustite formatiranje se nekim velikim brojem traka (recimo 90) i slušajte pažljivo zvuke disk-jedinice – trebalo bi da negde iza 80 traka čujete kako glava udara u graničnik. To će vam dati vrednost približnu maksimalnoj mogućoj. Zatim ponovo pokušajte formatiranje umanjujući svaki put broj traka za po jednu ova dok formatiranje prođe bez greške i bez sumnjivih zvukove iz disk jedinice.

Mi smo na ovaj način bez problema dobili 84 traka tj. 5 % više nego standardno. Ipak vam savetujemo da za sva kretnevnu upotrebu ograničite broj traka na 82 naročito ako diskete treba da prenosite na drugi računar.

Neki proizvođači (npr. sam IBM) tvrde da pokušaji formatiranja velikog broja traka mogu čak oštetiti disk jedinicu. Ova upozorenja su vanvrednе preterana (nama se nikad nije desilo nešto slično a svašta smo pokušavali) ali ih svakako imajte na umu ukoliko želite mnogo da eksperimentišete. Ako nešto zabrljate mi nismo krivi. ☺

MAKSIMALAN KAPACITET

Dolaze koji se postižu kombinovanim povećanjem broja traka i broja sektora po traci su zaista značajni. Maks maksi kapaciteti koji se mogu izvući zavise od tipa diskela i izno se

• 820 Kb za 5.25"	DD diskete
• 1.48 Mb za 5.25"	HD
• 1066 Kb za 3.5"	DD
• 1.72 Kb za 3.5"	HD

Sve ove je moguće pod uslovom da imate disk jedinicu visoke gustine (HD) tj. 5.25" deklarisanu na 1.2 Mb ili 3.5" od 1.44 Mb. Čak i ukoliko imate neki stari XT i njegovu DD-disk jedinicu od 360 Kb moći ćete bez problema da formatirate diskete na 410 Kb tj. oko 14% više.

Procentualno najveći dobitak (128% !) je se 5.25" DD disketama. Ove ekstremna vrednost proizlazi iz činjenice da IBM-ov standard od 360 Kb nije bio uslovljen kvalitetom diskete već mehanizmom disk-drajvova koji su se pravili u vreme kad je standard postavljen. Ta disk jedinice su kori stile traka dvostruko šire od današnjih tako da je na običnu disketu stajalo svega 40 traka.

U vreme postepenog prelaska se XT na AT računare javljali su se problemi pri čitanju 360 Kb disketa formatiranih na nekom AT-u (HD) u XT (DD) drajvovima. Do problema je dolazilo zbog toga što HD drajv formatira DD diskete na 40 traka CMD/NC CMD/NMC tako što formatira svaku drugu a na praviv treke dvostruke širine kao što to radi DD drajv.

Znači vaše 5.25" DD diskete ćete najbolje iskoristiti ako ih formatirate se 82 traka 10 sektora po traci. Njihov ukupan kapacitet biće po formuli (2) jednak 82 * 10 = 820 Kb. Sa programom FMT to se postiže ovako

```
FMT A: /T:82 /N:10
```

Ili A zamenite se B ukoliko imate lakvu konfiguraciju. FMT ima ugrađene skraćenice koje zamenjuju /T i /N parametre za neke najčešće korišćene formate. Za format od 820 Kb ta skraćena opcija je /F:820 (ukoliko koristite *FDFormat* te dvotačka između F i broja izlobaja nije potrebna dakle /F:820). Znači gore navedene komanda se može svesti na

```
FMT A: /F:820
```

Mi camo često koristili neskraćeni format komandi. Iz navedene broja traka i sektora poišo je on kompatibilan sa sva tri programa – koriste ga i *800* (odnosno DOS-ov FORMAT.COM) i *FDFormat* i *FMT*.

Velike (5.25") HD diskete se umesto 1.2 Mb mogu for matirati maksimalno na 1.48 Mb sa 82 traka i 18 sektora. Za ove diskete posebno je značajna mogućnost formatiranja na 1.44 Mb (80 traka 18 sektora) što je format identičan onome standardnom za 3.5" HD diskete. Tada se pot puno regularno obavlja kopiranje *(diskcopy)* malih (3.5") diskata ne velike. Ukoliko imate neki softver isporučen same na 3.5" a nemate dovoljne malih diskela za *backup*, onda se možete poslužiti 5.25" HD disketama formatiranim na 1.44 Mb. Pri lome na treba zaborav ti da su „velike" diskete jeftinije od „malih" što znači da ćete od ovakvih operacija izvući i neku finansijsku korist. Formatirane diskete na 1.44 Mb se obevlja komandom

```
FMT A: /T:82 /N:18
```

pri čemu opet ovo /T:82 /N:18 možete skratiti na /F:1.44 ili /F:1440

I kod 3.5" diskela su dobici nestandardnim formatiranjem značajni. Ukoliko ste nekad kao mali mali *Asan ST* ili *Commodore Amiga* računajka pa ste porasli i prešli na PC onda vam je sigurno ostala gomila 3.5" DS/DD disketa. MS-DOS ih koristi kao 720 Kb ali ih bez problema možete for matirati se 82 traka i 13 sektora na kapacitet od 1066 Kb. Ako slučajno budete imali problema se ovim formatom (a kod većine drajvova ne bi trebalo) smanjite broj sektora na 12 i dobićete ukupno 984 Kb.

Najviše što možete da izvučete na neko standardnoj diskel (ne računamo novi format od 2.88 Mb) je 1.72 Mb na 3.5" HD diskel (DOS je koristi kao 1.44 Mb). Ovo je zaista popriličnо prostora (20% više) i može bri vrlo korisno za *backup* većih količina pokatka. Ovaj kapacitet dobija se sa 82 traka i 21 sektorom

```
FMT B: /T:82 /N:21
```

U Tabel 2 prikazane su osnovne informacije o nekim nestandardnim formatima diskela, zavisno od veličine diska (5.25" ili 3.5") i vrste diskete...odnosno gustine magnetnog medijuma (DD ili HD). Uz formatiranu kapacitet prikazani su broj traka i sektora (za korišćenje /T i /N opcija) i skraćena oznaka koja se može koristiti za formatiranje pomoću programa *"MT*

Ukoliko želite malo da se igrate sa raznim kapacitetima disketa možete zadati bilo koje kombinacije /T i /N parametara nezavisno od vrednosti prikazanih u tabeli – treba samo da pazite da na prekoračite maksimalno dozvoljen broj traka i sektora za određeni tip diskete. Take npr. se 64 traka i 8 sektora možete dobiti potpuno egzotičan format od 512 Kb ili nake još čudnije. Od ovoga nećete mati neke praktične koristi ali možete obično zbunti nekoga ko nije upućan u problematiku.

JOŠ MALO PROSTORA

Slobodan prostor na tek formatiranoj disketi će uvek biti manji od njenog ukupnog kapaciteta. Ove stega što DOS jedan deo rezervise za svoje potreba (*system area*). Ovaj sistemski prostor se nikako ne može ukloniti (onda disketa ne be bila čitljiva iz MS-DOS-a) ali se može smanjiti.

DOS rezerviše prostor za datoteke koje se smeštaju u osnovni (*root*) direktorijum. Podrazumevani (*default*) maksimalni broj datoteka koje se mogu smestiti u *root* iznosi 112 datoteka za DD (sipod 1 Mb kapaciteta) i 224 za HD diskete (preko 1 Mb). Ovaj prostor je alociran nezavisno od stvarnog broja datoteka koje se nalaze u osnovnom direktorijumu. Čak i na tek formatiranoj potpuno praznoj disketi (na računajući fiktivne direktorijume i koj se automatski kreiraju) alociran je prostor za svih 112 ili 224 datoteke.

Prostor koji se ovim zauzima iznosi 32 bajta po datoteci tj. ukupno 7 Kb za svih 224 (na HD diskeli). Nalrgled nije mnogo ali obično je glavnine ovog prostora neiskorišćena. Ukoliko sla dobro organizovali smeštaj svojih podataka na diskelama onda obično na svakoj od njih imate svega nekoliko spakovanih arhiva. U tom slučaju možete bez problema smanjiti maksimalan broj datoteka u osnovnom direktorijumu i time uštedeti bespotrebno alociran prostor.

FMT dozvoljava zadavanje ovog maksimalnog broja opcijom /R (kod *FDFormata /D*) – *root directory* size pri čemu taj broj može biti bilo koji između 1 i 240. Pošto se prostor za *root* alocira u sektorima (512 bajta) ova vrednost biće zaokružena na prvi sledeći umnožak broja 16 (je je 512 bajta / 32 bajta po datoteci = 16) dakle realne moguće vrednosti su 16, 32, 48, 64 itd.

Na primer ukolike HD disketu formatsala evako

```
FMT A: /R:32
```

u osnovni direktorijum će moći da se smesti svega 32 datoteke (sasvim dovoljno ako držite sve arhivirano) *root* će zauzimati svega 1 Kb umesto standardnih 7 Kb

Tabela 1 Svi DOS formati disketa

Tip diske	Formatirani kapacitet	Potrebno od DOS verzije	Broj strana	Broj traka/sektora
5.25	160 Kb	1.0	1	40/8
5.25	320 Kb	1.1	2	40/8
5.25	180 Kb	2.0	1	40/9
5.25	360 Kb	2.0	2	40/9
5.25	1.2 Mb	3.0	2	80/15
3.5	720 Kb	3.2	2	80/9
3.5	1.44 Mb	3.3	2	80/18
3.5	2.88 Mb	5.0	2	80/36

BRZI RAD DISKETA

Osnovna namena programa *FMT* i *FDFormat* je formatiranje diskete na veći kapacitet, ali oni poseduju još jednu vrlo značajnu opciju koja dovodi do ubrzanja rada sa disketama. Ova tehnika zove se *sector sliding* (ili *spanning*) i može ubrzati rad sa diskom do 100%! U čemu je stvar?

Zamislite standardni disk od 360 Kb. On ima 9 sektora po traci koji su numerisani brojevima od 1 do 9. Glava disk-jedinice čita sekvencijalno (redno) ta sektora na jednoj traci. Međutim, za pomeranje glava na sledeću traku potrebno je malo vremena, taman toliko da joj „pobegne" prvi sektor na susednoj traci. Kada glava pređe na drugu traku već će se nalaziti na sektoru 2 ili 3 (zavisno koliko joj je pobeglo), jer se disk nepresta no vrti. Pošto čitanje (ili pisanje) podataka mora početi od prvog sektora, glava mora da čeka skoro pun jedan okret diska dok se opet ne pozicionira na prvi sektor.

Ovo se dešava ukoliko je *sector sliding* (bukvalan prevod je klizanje sektora) jednak nuli, što je standardno za DOS format. Ukoliko je *sector sliding* jednak jedinici onda je na prvoj traci redosled sektora „1 2 3 4 5 6 7 8 9" ali na drugoj se rotira (klizna) udesno za jedan pa je redosled „9 1 2 3 4 5 6 7 8" i tako za svaki sledeći sektor. Ako je *sector sliding* 2 onda se svaki sledeći rotira za dva itd. Znači, kada se glava prebacuje na sledeću traku i pri tom prebacivanju propusti jedan sektor, biće pozicionirana baš tamo gde treba i može nastaviti rad bez čekanja.

Standardni *sector sliding* se može promeniti pri formatiranju opcijama */X n* i */Y:n*. X znači da se pomen n sektora kada se promeni glava (strana diskete), ali na i traka dok Y znači da se pomen m sektora kada se pređe na sledeću traku. Obično samo Y opcija dovodi do konsnih poboljšanja, ali na našim konfiguracijama (naročito XT-ovima) i X može dovesti do daljnjeg povećanja brzine rada.

Dobra vrednosti parametra X su od 0 do 2, a za Y 0-4. Optimalne vrednosti za svoju konfiguraciju možete pronaći eksperimentišanjem, ali ako nemate vremena obično je */X:0 /Y:2* dobro rešenje. Autor *FDFormat*-a predlaže sledeće vrednosti zavisno od formata diskete:

- **320 – 410 Kb** => */X:1 /Y:3*
- **720 – 820 Kb** => */X:1 /Y:2*
- **1.2 – 1.72 Mb** => */X:2 /Y:3*

I mi smo merenjem ustanovili da su ovo optimalne vrednosti a tim što nismo primetili neke dobitke primenom */X* opcije.

Autori programa *FDFormat* i *FMT* navode da se koristćenjem tehnike *sector sliding* rad sa disketom ubrzava od 50 do čak 100%. Mi smo merenjima ugiavnom dobijali vrednosti oko 30% bolje od standardnih ali i to je vrlo značajno u obzirom na notornu sporost disketnih jedinica.

Ove nije sve što pruža program *FMT* (u većini slučajeva to može i *FDFormat*). Treba još napomenuti da se diskete nestandardnih formata mogu napraviti i kao sistemske ali ovo je obično nešto kompliikovanija operacija nego standardno DOS-ove **FORMAT /S**. Prvo je potrebno napraviti kopiju *boot* sektora sistemske diskete formatirane iz DOS a za šta služi poseban program FMTBOOT. Posle toga *FMT* tako dobijenu datoteku (*disk image*) koristi prilikom kreiranja sistemske diskete nestandardnog formata.

Još jedna zgodna stvar koju *FMT* i *FDFormat* rade je promena standardnog *boot* sektora ukoliko disketa nije sistemska. Ako se u trenutku dizanja sistema u disk drivu A nalazi ne-sistemska disketa sa standardnim (kako ga na previ DOS formati) *boot* sektorom ispliuće se poruka *Non-System disk or disk error*. Ukoliko nemate neku sistemsku disketu pri noći tada vam preostaje samo da izvadite disketu iz drajva i resetujete računar, pa će se sistem dići sa C diska. Ukoliko je disketa formatirana programima *FMT* ili *FDFormat* ona ima poboljšani *boot* sektor koji će u ovom slučaju samo javiti poruku upozorenja da disketa nije sistemska i odmah nastaviti dizanje sistema sa C diska. Ta ko se više nećete nervirati ako slučajno zaboravite disketu u drajvu A i resetujete računar.

Nestandardni formati u Tabeli 2 su dobro provereni u praksi i mogu su sasvim pouzdano koristiti

Tabela 2 Nestandardni formati

Tip diske	Formatirani kapacitet	Gustina medijuma	traka (T)	Broj sektora (N)	FMT opcije
5.25	360 Kb	DD	40	9	/F 360
5.25	400 Kb	DD	40	10	
5.25	410 Kb	DD	41	10	/F:410
5.25	720 Kb	DD	80	9	/F 720
5.25	800 Kb	DD	80	10	
5.25	820 Kb	DD	82	10	/F:820
5.25	1.2 Mb	HD	80	15	/F:1.2, /F 1200
5.25	1.36 Mb	HD	80	17	
5.25	1.44 Mb	HD	80	18	/F 1.44 /F:1440
5.25	1.48 Mb	HD	82	18	/F 1.48 /F 1476
3.5	720 Kb	DD	80	9	/F 720
3.5	820 Kb	DD	82	10	/F:820
3.5	984 Kb	DD	82	12	/F 984
3.5	1066 Kb	DD	82	13	/F 1.07 /F 1066
3.5	1.44 Mb	HD	80	18	/F 1.44 /F 1440
3.5	1.6 Mb	HD	80	20	
3.5	1.72 Mb	HD	82	21	/F 1.72, /F 1722

KAKO IZVUĆI NAJVIŠE

Kao što vidite, rad sa programima za maksimalno iskorišćenje diskete uopšte nije komplikovan. U većini slučajeva dovoljne će vam biti opcije */T* i */N* (ili skraćeni oblik */F*) za veći kapacitet i */X* i */Y* za brži rad diskete. Na treba zanemariti ni */R* za određivanje veličina osnovnog direktorijuma za još malo dobitaka u prostoru. Kad sve ovo spojimo dobićemo oblik komande za formatiranje nalik na ovaj:

FMT A: /F:1.48 /X:2 /Y:3 /R 32

Ovo su optimalni parametri (se izuzetkom */R* koji zavisi od toga za šta će vam disketa služiti) za formatiranje 5.25" HD diskete na kapacitet od 1.48 Mb sa *sector sliding*-om koji dovodi do najvećeg ubrzanja rada i sa alociranim prostorom za 32 datoteke u *root* direktorijumu (dodatnih 6 Kb dobitka ni račun sistemskog prostora). U slučaju 5.25" DD diskete (onih koje DOS formatira na svega 360 Kb) koman da bi bila ovakva

FMT A: /F:820 /X:1 /Y:3

pri čemu smo izbacili opciju */R* da vas ne zbunjuje – najbolje je da je sami postavite prema potrebama. I konačno ovo kako se najviše može izvući iz 3.5" HD diskete (nismo imali ni jednu 3.5" DD pri ruci da i za taj slučaj odredimo optimalni *sector sliding*):

FMT B: /F:1.72 /X:2 /Y:3

Ukoliko vam je sve ove kompilkovano za pamćenje FMT ima mogućnost da standardnu (*default*) kombinaciju koju koristite smesti u konfiguracionu datoteku. Naziv ove datoteke je *CONFIG.FMT* a on formatiranju *FMT* pokušava da je nađe u tekućem direktorijumu ili onom iz kojeg je startovan FMT.EXE. Možete podesiti i *environment* promenljivu FMTFILES koja ukazuje gde se nalaze datoteke koje FMT koristi npr. ovako

SET FMTFILES = C:\BIN\FMT

Svaki red CONFIG.FMT datoteke mora započinjati brojem „3" ili „5" – ovo označava da li se opcije u tom redu odnose na 3.5" ili 5.25" diskete. Možete koristiti „;" za komentare. Eve jednog primera konfiguracione datoteke

```
5 DEFAULT /T:82 /N:18 /X:2 /Y:3 /A
3 DEFAULT /T:82 /N:21 /X:2 /Y:3 /A
5 1530          /T:85 /N:18
```

Ovim smo definisali standardne vrednosti za formatiranje 5.25" i 3.5" diskuti (prvi dva reda) – maksimalni kapacitet i optimalni *sector sliding* dok opcija */A* služi za ispisivanje dodatnih tehničkih informacija o disketi. Preporučujemo vam da */A* stavite u vašu konfiguraciju – višak informacija nikad na škodi a standardna poruka samo o ukupnom kapacitetu formatirane diskete nije previše informativna.

Poslednjim redom (**5 1530 /T:85 /N:18**) definisali smo skraćenicu za novi format od 1530 Kb – šada će biti dovoljno kucati **FMT /F 1530**. Ovaj format koristi svih 85 traka na disketi zato sa sebe nabeg upozorenja o opasnu kad radite se više od 82 traka.

I *FDFormat* može imati svoju konfiguracionu datoteku (FDFORMAT.CFG) ali su u njejovom slučaju mogućnosti podešavanja znatno skromnije – jedina korisna stvar koja se može definisati je standardni *sector sliding*.

POUZDANOST

Čemu sva ova poboljšanja ako sve ne radi dovoljno pouzdano? Ne treba da brinete – svaška smo pokušavali i niemo imali nikakvih problema se pouzdanošcu rada (što ne znači da i dalje nismo oprezni – važne podatke spašavamo i dalje držimo na „običnim" disketama, iako za ovo više nemamo nekih dobrih razloga).

Sve diskete koje su bez problema (znači bez jednog lošeg sektora) prošle standardni DOS-ov FORMAT bez problema su radile i sa većim kapacitetima. Ak dešavalo se da naka disketa standardno formatirana imaju loše sektore a kada se to uradi sa *FMT* ili *FDFormat*, oni ipak da je sve Ok. **Pazite – FMT i FDFormat imaju slabiju kontrolu ispravnosti diskete!** Kada smo proveravali da li je u redu lokva nazovi-ispravna disketa (npr. preko Nortonovog *Disk Doctor*-a) ustanovili smo da loši sektori i dalje postoje samo ih programi nisu uvek javljali kako treba.

Ovo mana je naročito nepojmna kod *FDFormat*-a koji ima katastrofalno lošu provenu ispravnost. **FMT** to radi znatno bolje praktično kao i DOS ali nam je i on zabrljao dva ili pura. Ne budite sigurni čak ni u DOS-ovu komandu za formatiranje jer nam je i ona neki put napravila probleme. Upravo je bolje kontrolisi ispravnosti jedan od najvećih razloga zbog kojih koristeme **FMT** umesto *FDFormat*.

Ako imate diskete dobrog kvaliteta neka vas ovo ne brine. Korišćenjem opcije */V (verify each disk write)* prilikom kopiranja odnosno *COPY /V* nećete imati potrebu za dodatnim proveravanjem ispravnosti. Ovo će uspeti kopiranje ali pouzdanost je ipak važnija.

Da ponovimo još jednom – diskete formatirane pomoću *FMT*-a neki veći kapacitet (u okviru granica iz naše Tabele 2) su **jednako pouzdana** kao i one formatirane DOS-ovom komandom FORMAT. Treba samo biti vrlo oprezan kad imh diskeia za koja znate da imaju loših sektora, ali to je uobičajeno – i ovako vam na bismo savetovali da takve diskete uopšte koristite za ikle važne stvari.

DA LI SE ISPLATI?

Da rezimišemo – bez mnogo muka dobijete diskete **od 20%** da **130% veću kapacitetu** uz **ubrzanje rada od 30 do 100%**, **nesmanjenu pouzdanost**! Biti bate zaista ekstremno konzervativni kada na biste iskoristili do maksimuma ono što možete, a ništa vas na košta.

Čak i bukvalno zaista vas ništa na košta. Sva ti programa o kojima smo pisali su potpuno besplatni, pošto spadaju u kategoriju *public domain* (u javnom su vlasništvu) odnosno *freeware* programa. Sve što vam je potrebno mešta za nekoliko minuta pokupiti sa Sezama iz direktorijuma R:\ISM#C\DISKUTIL\. Od ih programa (800 i FDFormat i FMT) **savetujemo vam da uzmete FMT** njegove prednosti smo već opisali u tekstu.

Napravimo malu računicu. Jednu 5.25" HD disketu standardnim DOS postupkom formatirate na 1.2 Mb što znači da u jednu kutiju (10 komada) disketa može da stane 12 Mb podataka. Ali ako ih pomoću programa FMT (ili nekog drugog) formatirate na 1.48 Mb (ili čak vas ukoliko vaša disketna jedinica to može da izdrži) moći ćete da smirite ukupno 14.8 Mb podataka. Znači, samo na jednoj kutiji ušteđeli ste skoro 3 Mb odnosno praktično ste dobili besplatne dve prazna diskete!

Ako se još uvek pitate da li se isplati – naravno da se isplati i nemojte više lupati glavu oko toga!

NIKO KAO COREL

Kao grom iz vedra neba, bez ikakva najava, početkom leta pojavila se nova verzija programa *CorelDraw*, sa oznakom 4.0, koja ovaj već legendarni program definitivno učvršćuje na vrhu top-liste profesionalnih alatki za grafički dizajn. Ovaj prikaz predstavljva svojevrsnu dopunu umetka iz „Računara" 93, u kome je *Corel* predstavljen do najsitnijih detalja.

Miroslav Jovanović

Ako se verzija 3.0 shvati kao verzija koje donosi novine vezane za nešto konformni rad (*Roll-up* meniji) i pobojšanje vezana za editovanje teksta (*Paragraph text* rad sa *True Type* slovima) za ovu verziju se može reći da donosi sasvim novi pristup obradi kompleksnog teksta (tekst + slika). Osim promene na samom *Corelu*, došlo je do izmena i u programima koji čine komplet programskog paketa. Znatno su poboljšani postojeći programi u pogledu pojedinih mogucnosti i konfora rada kao i kompatibilnosti sa *Corelom (CorelCHART CorelPHOTO-PAINT CorelTRACE CorelSHOW* i *CorelMOSAIC)* Novi član porodice je *CorelMOVE* program koji vam omogućuje kreiranje kompjuterske animacije

Naravno treba reći i to da *Corel 4.0* traži obavezno *Windows3.1* a što se hardvera tiče još uvek je moguće raditi i na 286 mašinama (traba imati nešto veći RAM za tole konformiji rad) Koncepcija rada bez matematičkog koprocesora je i dalje zadržana

EKRAN COREL-a 4.0

Nakon instaliranja i aktiviranja programa, uočićete da je izgled ekrana u osnovi isti I dalje je prozor za crtanje ostao uokviren standardnim delinama za *Windows* okruženje Razlika koja se bomah uočave je novi meni **Layout.** Ovaj novi meni je nasledio mesto menija **Transform,** koji je „raseljen" po ostalim menijima. Meni **Layout** objedinjuje komande vezane za rad sa kompleksnim dokumentom Tu su sekcije za rad se stranicama (dodavanje brisanje definisanje listanje strane) za rad sa slojevima (**Layers Roll-Up**) za rad sa stilovima (**Styles Roll-Up**) i za rad sa **Grid** i **Guidelines** komandama

Novine ove verzije je i to de kompleksnije alatke nije više nepomičan na ekranu Kroz **Display** meni je dale mogućnost da alatke definišete kao **Floating Toolbox** i kao takve da ih pomerate po ekranu a ako treba i da ih uklonite

Ukoliko definišete da vaš dokument sadrži više bo jednе strane onda vam se u okviru skrol-dara za pomeranje prozora za crtanje po horizontali javljaju i dodatne opcije za listanje stranica. Možete da definišete najviše 999 stranica u okviru jednog dokumenta

Corel 3.0 je kao novinu doneo *Roll-Up* menije Može se reći da verzija 4.0 obiluje *Roll-Up* menijima (seda ih je ukupno 14) Oni su zanimljivi jer objedinjuju više komandi mogu se pomerati po ekranu da ne smetaju pri radu a mogu se i isključiti kada nisu potrebni

COREL 4.0 I ALATKE

Alatke su u osnovnom prikazu ostale iste a samo je kod pojedinih došlo do razdvajanje predloženih opcija.

SHAPE TOOL – alatka oblika je preirpela manje izmene U **Node Edit Roll-up** meniju je natpis na tasteru **Add** zamenjen natpisom + **Delete** natpisom – i **Join** je zamenjena sličicom spojenih kanke lanca dok je **Break** zamenjen sličicom prekinute kanke lanca. **Node Edit Roll-up** meni je dopunjen sa tri komande **Auto Reduce** – omogucuje da se ekstremne tačke na krivama (dobijene trejsovanjem) eliminišu čime se znatno olakšava rad **Elastic Mode** – je novi način rada sa čvorovima, e uključenje ove opcije omogućuje pomeranje više odabranih čvorova, zavisno bo njihovog položaja u odnosu na čvor koji ste zahvatili prilikom pomeranja a ne keko je to do sada bilo – linearno **Pressure Edit** – ova opcija editovanja čvorova omogućuje da po želji menjate atribute **PowerLines** opcije pomoću **Node Edit Roll-up** menija.

PENCIL TOOL – Ako zadržite pritisnut taster miša malo duže prilikom izbora ove alatke pojaviće vam se pored vec postojećih ikonica za izbor opcije **Freehand** i **Bezier** načina rada i tri nove ikonice za rad sa kotama nacrtanih objekata. Kole možete da poslavite – nacrtate i upišete dimenziju nacrtanog objekta u željenom lontu i jedinici mere

TEXT TOOL – Slično kao i kod prethodne alatke ako zadržite pritisnut taster miša, pojaviće sa dodatne ikonice Nova ikonica je za rad sa paragref-lekstovima Sada je moguće u jedan paragraf smesliti do 4000 karaktera što je sasvim dovoljno za celu stranu teksta. Novina je i to de možete menjati oblik *envelope* tekst-paragrafa. Rad pri obradi teksle je poboljšan i dodavanjem *indents* i *bullets* komandi **Symbol** alatka je dobila svoj **Roll-Up** meni koji se aktivira izborom ikonice za simbole iz ove alatke Željeni simbol sada ne crtez postavljate zahvatanjem sa ekrana **Roll-Up** menija i postavljanjem u prozor za crtanje

FILL TOOL – Ove alatka je u osnovi ostala ista, s tim da su dodate nove opcije Rad sa teksturama omogućuje upotrebu lekstura pojedinih materijale kao boje površine objekta što u prethbonoj verzji nije bilo omogućeno Nova mogucnosi je i to de površinu objekta pored radijalno i linearno postavljenog prelaza obojte najsamsama i sa konusnim prelazom Sada je moguće definisati i više prelaza u okviru samo jednog objekta polom medu-boju u prelazu a definisan prelaz po želji možete da snimite i da ga kasnije koristite i na drugim objektima

LINIJA MENIJA

Kako je već rečeno, u linji menija je došlo do promene Najuočljivije promena je pojava novog **Layout** menija i nestanak dosadašnjeg **Transform** menija Ali podimo redom bo menija do menija

FILE – Komanda **Save As** dobila je u okviru svog dialog boksa opciju **Save as CorelDRAW Template** (*.CDT) Ova novina je u skladu sa tendencijom da u *Corel* budu objedinjena i funkcija crtanja i editovanja teksla Komenda **New From Template** omogućuje da prethodno pripremljenu matricu dokumenta učitate i konstile u okviru novog dokumenta koji radite Komande **Import** i **Export** su dobile obnovljene liste uvoznih filtera tako da se proširuje kompatibilnosi *Corela* sa ostalim postojećim softverom **Insert Object** komanda je proširena mogućnošću da drugi fajl ubacite *(insert)* kao objekat u postojeci dokument.

Komanda **Print** je jedna bo komandi koje su doživele radikalne promene Aktiviranjem ove komande dobićete sasvim novi dijalog-boks Upečatljivo mesto zauzima prostor sa pregledom stranica za štampanje u kojam možete menjati i veličinu crteža pomerati ga po prikazanoj stranici Druge polovina **Print** dijalog boksa praktvena je sada mnoštvom novih opcija i komandi Pri samom vrhu imate mogucnosi da definišete štampač na kojem želite da štampate a bomah pored natpisa sa izabranim štampačem imate mogućnost pregleda njegovih grafičkih mogućnosti Značajan deo zauzima opcija **Pages,** koja omogućuje izbor štampanja svih stranica ili stranica od– do (nedostaje mogućnost štampanje parnih ili neparnih strana kao i izbor pojedinih stranica koje nisu u nizu) Komanda **Colors** omogućuje da štampate vaše crteže sa separacijom boja Onaj ko je imao potrebu da sa dosadašnjim verzjama radi pripremu za štampu u više boja sigurno će se obradovati ovoj opeji Pnlikom štampanje sa **Print to File** seda imate mogućnost da fajl pripremite za stampanje i sa Macintosh kompjutera. Komanda **Options** vas upućuje ne dodatni dijalog boks sa čitavim nizom parametara za kontrolu i način štampanja vašeg crteže (**Fountain Steps, Print Negative, Emulsion Down** i dr)

Komanda **Printer** omogućuje da podesite pojedine parametre samog štampača (dijalog-boks vec poznat iz prelhodne verzije) Posebari prostor u okviru dijalog-boksa zauzimaju komande **Position and Size** Pruža vam se mogućnost da isključite pregled stranice za štampanje. da aktivirate komandu **Fit to Page** da aktivirate komandu **Tile** ako sa **Scale** povecate crtež toliko da on ne stane na jednu stranicu podešenu na štampaču da aktivirate komandu **Center** koja postavlja vaš crtez na sredinu stranice da sa komandama **Width** i **Height** podesite veličinu crteža da sa komandama **Left** i **Top** definišete tačan polozaj gornjeg lsvog ugla crteža u bonosu na postavljeni **Grid Origin** da sa **Scale** definišete koliko će pn štampanju vaš crtež biti umanjen ili uvećan

EDIT – Komande **Undo** i **Redo** sada imaju mogucnost definisanja više koraka Komanda **Clone** kopira boabran objekat (objekte) sa definisanim pomakom, eli tako de ostaje veza izmedu ovih kopija koja omogućuje da sve izmene i transformacije učinjene na polaznom objektu *(master)* budu automatski izvedene i na završnom *(clone)* objektu **Copy Attributes From** omogućuje prenos atributa na izabrani objekat (objekte) sa nekog objekta (**Outline Pen, Outline Color, Fill, Text Attributes** i **PowerLine** atributa) Komanda **Edit Text** je prebačena u meni **Text.**

LAYOUT – Ovo je novi meni sačinjen bo novih komandi i komandi koje su se nalazile u prethodnoj verzji ali u drugim menijima Nove su **Insert Page** – umetni stran cu otvarajuci loko višestranicnu dokument **Delete Page** – bnše definisanu stranicu iz višestranicnog dokumenta **Go To Page** – prikazuje definisanu stranicu u višestraničnom dokumentu. Koman da **Page Setup** je stara komanda koja je dobila novi izgled i opcije vezane za rad sa višestraničnim dokumentima Vrlo zanimljiva novina je mogućnost da prilskom na samo jedan taster podesite veličinu stranica koje je definisana za vaš štampač Komanda **Styles Roll-Up** aktivire odgovarejući **Roll-Up** meni pomoću kojeg možete konstili postojeće matrice dokumenata ili kreirati nove Poznata komanda **Layers Roll-Up** je sada dobila i mogućnost da definišete **Master Layer** koji sadrzi objekte koji ce sa pojavljivati na svim stranicama višestraničnog dokumenta. Od komandi koje su postojale i u ranijoj verzji ovaj meni sadrži **Grid Setup** – proširen komandom **Scale,** koja omogucuje da definišete vezu izmedu stvarnih dimenzija vašeg crteža i dimenzija prikazanih u Corelu **Snap To** sada objedinjuje komande **Snap To Grid, Guidelines** i **Objects.**

EFFECTS – U ovom meniju se sada nalaze komande **Rotate & Skew, Stretch & Mirror** i **Clear Transfor-**

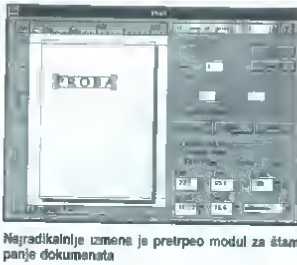
mations, koje su se u prethodnoj verziji nalazile u meniju **Transform**. Komanda **Envelope Roll-Up** aktivira novi *roll-up* meni, koji sada objedinjuje sve komande vezane za rad sa envelopom objekta iz prethodne verzije. Dodatna mogućnost je da možete da izaberete envelopu iz postojećeg seta envelopa kao i to da vaš objekat možete da pridružite envelopu u obliku odjekta kojeg nacrtate. Komanda **Extrude Roll-Up** je dopunjena novim unapred pripremljenim primerima. **Conture Roll-Up** je potpuno nova komanda i omogućuje vam da uz podešavanje više parametara dodate vašem objektu jednu ili više konturnih linija. **PowerLine Roll-Up** je takođe nova komanda pomoću koje definišete kako će izgledati linije prilikom crtanja. Postoje 24 unapred pripremljene oblika što ne znači da ne možete da definišete i oblik po vašoj želji. Komanda **Copy Effects From** je samo novi naziv za komandu koja objedinjuje dosadašnje komande **Copy Perspective** i **Copy Envelope From**. Kao i prethodne komande **Clear Effects** objedinjuje funkcije više već poznatih komandi (**Clear Perspective, Clear Envelope, Clear Blend** i **Clear Extrude**)

TEXT – **Text Roll-Up** meni je dopunjen mogućnošću da vidite deo teksta u odabranoj vrsti fonta. Taslen **Frame** i **Paragraph** su dodeli dok su tasten za **Bold** i **Italic** zamenjeni list boksom. Komanda **Paragraph** je doradena i sada nudi mogucnosti definisanja razmaka ispred i iza pojedinog paragrafa potavljanja tabulatora, postavljanja i definisanje oblika za *indents* i *bullets*. Mogucnosti rada se tekstom su proširene i komandame **Find** i **Replace**, komandama za pronalaženje i zamenu definisanih reči u otkucanom tekstu. Komanda **Edit Text** je samo prebačena iz menija **Edit**.

ARRANGE – I ovaj meni je drugačija osmišljen i dopunjen odredenim novinama. Komanda **Move**, ostala je ista kao u ranijoj verziji ali je prenesena iz menija **Transform**. Komanda **Order** objedinjuje komande **To Front, To Back, Forward One, Back One** i **Reverse Order**. Komanda **Weld** je nova i omogucuje sjedinjavanje više objekata. Zanimljiva je upotreba kod objekata koji se dalimično preklapaju. Tako dobijeni objekat ima okvirnu liniju koja u stvari opisuje siluetu tih odjekata a ima boju površine poslednjeg odabranog objekta.

DISPLAY – Komande **Grid Setup, Snap To Grid, Guidelines Setup, Snap To Guidelines** i **Snap To Objects** su izoslavljene iz ovog menija i prebačene u meni **Layout**. Komanda **Show Color Palette** je modifikovana novim opcijama (**Custom Palette, Pantone Spot Palette, Pantone Process Palette** i **Trumatch Process Palette**) Nova komanda je **Floating Toolbox**. Ona vam daje mogucnosti da alatke učinite pomičnim po ekranu a u krajnjem slučaju i da ih isključite

SPECIAL – Komanda **Create Symbol** je dodata ovom meniju i omogućuje vam da sami dodajete simbole u biblioteku simbola. U okviru ovog menija dodate su komande **Extract** i **Merge-Back** koje su u prethodnoj verzji Corela bile u okviru menija **Text**. Komanda **Preferences** je postojala i u prethodnoj verziji ali je u ovoj verziji dobila i neke značajne novine. Osnovni dijelog boks je dopunjen opcijama **Undo Lavels** – omogućuje da definišete koliko je mogućih *undo* nivoa tačnije koliko Korela izvršnih komandi se može poništiti. **Place Duplicates and Clones** – je već poznata komanda sa proširenjem delovanja i na komandu **Clone**, koje je novina ove verzije. **3.x Compatibility Message** – uključuje/isključuje prikom učitavanja crteža iz **Corel 3.x** verzije pojavljivanje dijalog boksa koji vam omogućuje konverziju spejsinga teksta između verzija 3.0 i 4.0. Taster **Dimensions** aktivira novi dijalog-boks u kojem mozete definisati kako će biti orijentisani natpisi (horizontalno ili vertikalno centrirano ili ne) prilikom crtanja kota a aktiviranjem i dodatnog dijalog boksa **Format Definition** možete definisati oblik ispisa dimenzije s obzirom na izabranu jedinicu mere. Iester **Roll-Ups** omogucuje da definišete pojavljivanja **Roll-Up** menija – od mogucnosti da se prilikom aktiviranja Corela aranžiraju i pojave svi



Najradikalnije izmene je pretrpeo modul za štampanje dokumenata

Roll-Up menji do mogućnosti da se na pojavi ni jedan

OBJECT – Ovaj meni je potpuno nov na nalazi se u liniji menija a aktivira sa klikom na objekat desnim tasterom miša. Sadrži komande **Save As Style – aktivira** dijalog-boks koji omogućuje da snimite stil zasnovan na atributima selektovanog objekta. **Update Style** – aktivira dijalog-boks za snimanje izmena nekog vec postojećeg stila. **Revert To Style** – obnavlja atribute objekta koji su definisani stilom izabranim za dati objekat. Ako je na njemu bilo izmena u odnosu na definisani stil. **Apply Style** – aktivira spisak postojećih stilova sa kojeg mozete odabrati stil koji ćete pridružiti vašem objektu. **Overprint Outline** i **Overprint Fill** – omogućuju rad sa separacijom boja kod pripreme za štampu u više boja. **Data Roll-Up** – aktivira meni za rad sa bazom podataka o objektima koji mogu biti brojčani (numerički) tekstualni vremena i datumi. **Select Master** – selektuje odjekal od kojeg je dobijen selektovan *clone* objekat. **Select Clone** – selektuje odjekte koji su dodjeni od selektovanog master objekta. **Revert to Master** – *clone* i *master* objekat su vezani sve dok na *clone* objektu ne promenimo neki od atributa a ova komanda omogućuje da *clone* objekat ponovo vežete za master po pojedinim il svim atributima, koji se definišu kroz poseban dijalog-boks

PROGRAMI IZ PAKETA

Na prvi pogled, gotovo sve je ostalo isto jedina uočljiva razlika je pojava programa za animaciju – *CorelMOVE*. Ako svaki od ovih programa malo bolje pogledate, primetićete da i pored u osnovi istih mogućnosti ovi programi sadrže i dosta novina. Kod nas je svim ovim programima uglavnom posvećivana vrlo male pažnja, što nikako ne odgovara njihovim mogućnostima

CorelPHOTO-PAINT je vrlo snažan program za kreiranje i obradu bitmapiranih slika. Da bi što više ličio na „člana porodice" program je opremljen nizom *roll-up* menija (**Color Selection Roll-Up, Tool Settings Roll-Up, Canavas Roll-Up, Fill Settings Roll-Up, Bitmap Textures Roll-Up**) Mnogo je učinjeno i u pogledu lakšeg ovladavanja ov m programom jer je način rada ujednačen sa samim Coralom. Seda je u okviru programa omogućena i upotreba skenera bez izlaska iz programa. Uključena je i upotreba raznih tipova novih četkica, razdvojeno štampanje boja i di

CorelCHART predstavlja dobru kombinaciju programa za unos i obradu podataka i piograma za njihov grafički prikaz. Jednom upisane podatke možete mesmonsali sortirati pregledati i obrađivati medusobno koristeći čak više od 45 unapred pripremljenih matematičkih i finansijskih funkcija. Ovako obrađene podatke možete prikazati grafički, vrlo jednostavnim odabiranjem jednog od načine prikaza z povećeg spiska. Uz sve ovo pruža vam se mogucnost da već grafikon ukrasite obiljem natpisa legendom a sve to podrzano svim *True Type* slovima, koja inače poseduje u okviru *Windows*-a. I ovaj program je pretrpeo više promena koje su za cilj imale da što više unificiraju rad sa programima iz programskog paketa Corel. **Print** dijalog boks je identičan onome iz Corela, alatke za rad su vizuelno identične a i koriste se gotovo isto kao u Corelu sa pojedinim specifičnostima vezanim za rad sa ovakvim programom

CorelMOVE je novina ovog programskog paketa i predstavlja program kojim ćete moći „dah život" vašim crtežima. Omogućuje da kreirate animaciju konsteći metodu „slika po slika". Slike možete napraviti u programime *CorelDraw* i *CorePHOTO-PAINT* i uvesti ih u *MOVE*. Animacija je novina ovog paketa i uvedena je varovatno da bi se mogao držati korak sa konkurencijom. S obzirom da je dosadašnja orijentacija programekog paketa bila usmerena u pravcu izrade crteža i integracije teksta, koliko će ovaj program biti korišćen pokazaće sa u praksi. Naravno nikako ne treba potcenjivati uloženi trud koji će verovatno doneti mnogo zadovoljstva onima koji kreiraju razne prezentacije na svom kompjuteru

CorelTRACE - Ovo je stari član porodice, ali u prilično doredenoj verziji. e omogućuje transformisanje bitmapiranih u vektorske slike, koje se dalje mogu obradivati u Coralu. I u ovom programu je mnogo učinjeno da vizuelni utisak podseća na Corel okruženje. Sada imate mogućnost da istovremeno na ekranu vidite bitmapiranu sliku i trejsirani, vektorsku sliku. Sliku snimate sa eps ekstenzijom tek naknadno a ne kako je to u prethodnoj verziji bilo istovremeno sa trejsiranjem. Velike novina je i mogućnost da trejsirate samo deo odabrane slike. Sada i ovaj program omogućuje direkino korištenje skenera za unos slike. Sliku je moguće učitavati i iz *CCapture* i *Hijack*. Programu su dodate i nove alatke. Sada vec neizbežni *roll-up* meniji

CorelMOSAIC - I ovo je program iz starog programskog paketa. Omogućuje pregled biblioteka slika. Način rada i izgled su vrlo malo doređeni

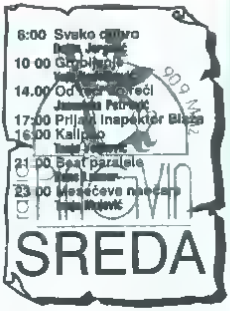
CorelSHOW je takode program iz starog paketa, ali i nešto doređen. Omogucuje da kreirate prezentacuju slika koje su pravljene u programima *CorelDraw CorelPHOTO-PAINT, CorelMOVE* i *CorelCHART*. Prilikom kre ranje prezentacije možete da definišete koliko će vremena svaka slika da bude na ekranu te razne načine pojavljivanja i nestajanja slika

CorelCAPTURE - Ovaj program omogućuje da „uhvatite" deo ili ceo ekran u kojem radite uz pomoc kombinacije tastera a da pri tome ne izlazite iz programa u kojem radite. Tako dobijene slika smešta se u **Clipboard**, odakle je uz pomoć **Clipboard Viewer**-a možete pogledati ili sa **Paste** komandom uneti u neki od *Windows* programa (*Word Corel Paintbrush CorelPHOTO-PAINT*...) kao bit map sliku. Ovaj program radi sasvim solidno sa svim programima iz *Windows* okruženja ali je znatno slabijih mogućnosti nego *Hi-Jaak*. Ne poseduje izbor filtera za konverziju slika iz različitih programa pa će se njime zadovoljiti verovatno samo uži krug korisnika.

UMESTO ZAKLJUČKA

Nakon razmatranja svih novina koje su se pojavile u verziji *CorelDraw 4.0* može se reći da ih ima dosta i da u velikoj men doprinose boljem iskorišcenju mogućnosti računara u odlasti grafičkog rada. Moguće je da će ova verzija naći i ne otpor kod jednog dela korisnike, naročito amatera, jer su se u njoj znatno više nego u prethodnim verzijama pojavili termini i načini rada svojstveni profesionalnim grafičkim radnicima. Znatno su poboljšane opcije za izradu predložaka za različite vrste štampe. I pored odrada čiji je cilj lakša i bolja obrada unesenog teksta kao i mogućnost da kreirate dokumenl od 999 stranica, malo je verovatno da će se neko odlučiti da radi prelom knjige pomocu ovog programe. Sve u svamu nova verzija je tu e vreme će pokazati koliko vredi

JEDINI PINGVIN KOJI USPEVA U TOPLIM KRAJEVIMA...



Info-D
Informatički inženjering

Tel /Fax (38 11) 613 822
Grodski park b b
11080 ZEMUN

KOMPONENTE	386SX 33MHz	386DX 40MHz	486DX 50MHz
OSNOVNA PLOČA,PROCESOR	386SX 33MHz	386DX 40MHz,128C	486DX 50MHz,256C
RADNA MEMORIJA	1MB	4MB	4MB
HARD DISK	85MB	125MB	211MB
FLOPI DISK	5 25"/1 2MB	5 25"/1 2MB	5 25"/1 2MB
KONTROLER (2x HDD/FDD)	IDE	IDE	IDE
I/O KARTICA	2 ser ,1 par	2 ser ,1 par	2 ser ,1 par
GRAFIČKA KARTICA	VGA 512KB	VGA 512KB	VGA 512KB
KUĆIŠTE	DESKTOP	MINI TOWER	BIG TOWER
MONITOR 14"	MONO VGA	MONO VGA	MONO VGA
TASTATURA (101 tipka)	USA klik	USA klik	USA klik
CENA	**1.460**	**1.950**	**3.190**

RAZLIKA U ODNOSU NA OSNOVNU KONFIGURACIJU ZA.

VGA Kartica 1MB	30	Desktop < - > Mini Tower	20	Hard Disk 85MB< >125MB	70
SVGA kolor monitor	350	Desktop < -> Big Tower	150	Hard Disk 85MB<- >211MB	290
Flopi Disk 3 5"/1 44MB	110	Mini Tower < -> Big Tower	30	Hard Disk 125MB<- >211MB	220
Flopi Disk 5 25"/1 2MB	120	Miš sa priborom	50	Memorija 1MB	120

" E P S O N " štampači	CENA
LX 100 YU(852 855)kabl,traktor	550
LX 400 YU, kabl	480
FX 1050 YU, kabl	1.100
LO 100 YU(852 855)kabl	580
LO 570 YU(852 855)kabl	880
LO 1070 YU(852 855)kabl	1.250
DFX 5000, kabl	3.960

MLAKAR & CO

AVSTRIJA

CHASSIS	DEM
CASE BABY & POWER S (+3.5)	86.80
CASE SLIM & POWER S	125 30
CASE SLIM 8801 & POWER S TUV	126.40
CASE MINI TOWER & POWER S	86 00
CASE TOWER & POWER S TUV	156 30

MOTHERBOARD	DEM
MOTHERBOARD 386/SX 33MHZ	149 00
MOTHERB 386-40MHZ, 128KB C	229 00
MOTHERB 486-33MHZ, 256K C VL	768 90
MOTHERB 486-50MHZ, 256K C VL	1 056 70
MOTHERB 486DX2-66MHZ, 256K C VL	1 210 50

DISPLAY CARD	DEM
MONOCHR -GRAFIC/PRINTER CARD	21 10
SUPER VGA CARD 1024×768/512KB	61 00
SUPER VGA C 1280×1024/1MB TSENG L. 64K C	139 00
S VGA C 1024×768/1MB TSENG L., VL	170 70
VGA CARD S-3 VL 1MB	246.40
VGA CARD S-3 VL 2MB	295 70

CONTROLLER	DEM
AT (IDE) BUS HDD/FDD CONTROLLER	18.30
AT (IDE) BUS HDD/FDD CONTROLL. + I/O	31 60
AT (IDE) BUS HDD/FDD+I/O VL DC2000	61 10
AT (IDE) BUS CACHE HDD/FDD CONTR 512K	244 00
AT (IDE) BUS CACHE CONTR CK VL	272 10
ADAPTEC SCSI-2 AHA 1542C KIT	432 00

ADD ON CARD	DEM
I/O CARD AT (SER PORT)	13 80
I/O CARD (PAR/2 SER/GAME PORT)	18 90
MULTI USER CARD (4×RS232)	79 80
MULTI USER CARD INTELLIG (8×RS232)	521 20
MULTI-USER CARD INTELLIG (16×RS232)	887 10
AD/DA CARD 12 BITS	84 30

LAN	DEM
ETHERNET COMPAT (NE2000) 8 16BIT	74 90
ETHERNET POCKET LAN ADAPTER	321 60

BAR CODE	DEM
BAR CODE READER MR500-1	284 30
CCD SCANNER MS-230 W/MR500-2	634 30

RAM	DEM
RAM MODULE 1MB×9/70NS SIMM	dnevna cena
RAM MODULE 4MB×9/70NS SIMM	dnevna cena

COPROCESSOR	DEM
COPROCESSOR 80387SX 33MHZ	117.30
COPROCESSOR 80387-40MHZ ULSI, IIT	140 00

KEYBOARD	DEM
KEYBOARD 101 MITSUMI/SAMSUNG	37 50
KEYBOARD 101 CLICK MINI	58 30
KEYBOARD 101 CLICK CHIODNY YU	53 20

FLOPPY DD	DEM
FLOPPY DISK 5.25" 1.2 MB MITSUMI	85 00
FLOPPY DISK 3.5" 1.44 MB MITSUMI	70 00
FLOPPY DISK 5.25" 1.2 MB PANASONIC	94 30
FLOPPY DISK 3.5" 1.44 MB PANASONIC	71 10

HARD DISK	DEM
HARD DISK ST 3144A 130MB/16MS	342 90
HARD DISK ST 3290A 260MB/16MS	478 60
HARD DISK ST 3550A 452MB/12MS	1 116.00
HARD DISK ST 3600N 525MB/11MS	1 623 40
HARD DISK ST 3600A 540MB/11MS	1.572 00
HARD DISK ST 11200N 1 050MB/11MS	2 292 00
RDD SYQUEST 5110	567 40

Računare prodajemo u kitu (u delovima). Za sve komponente nudimo garanciju, montazu i servisiranje u Srbiji Stručni savat pri izboru možete dobiti na telefon 9943/4227-2333 Naša prodavnica se nalazi u Austriji, u Podgori (Unterbergen), na putu za Celovec, 60 kilometara od Ljubljana i 12 kilometara od Ljubelja Radno vreme od 9—17, subotom od 8—12 Faks 9943/4227-2091

STREAMER	DEM
STREAMER 120/250MB INT (DJ 20)	318 60
STREAMER COLORADO TRACKER	742 30
CARTRIDGE CL2120 125/250MB	41 40

MONITOR	DEM
MONITOR 9" VGA MONOCHR	209 20
MONITOR 14" B/W MONOCHR.	139 70
MONITOR VGA 1024×768 0 28	457 10
MONITOR VGA MITAC 1024×768 LOW RADIAT	480 00
MONITOR VGA MONOCHROME	184 30
MONITOR VGA TVM MONO LOW RADIAT	195 70
MONITOR VGA 17" 1024×768 N I	1.218 70
MONITOR VGA 20" 1280×1024 N I	2.151 20
MONITOR PHILIPS 14" VGA COLOR	449 00
SCREEN FILTER MOND 14"	12 90
SCREEN FILTER CDLOR 14"	14 30
SCREEN FILTER GLASS 14"	18.00

PRINTER	DEM
PRINTER CITIZEN SWIFT 200	427 10
PRINTER C T I 9PIN A3	354 30
PRINTER STAR LC-15 (NX1500)	490 00
PRINTER HP LASERJET IV	2 850 00
PRINTER HP LASERJET IVL	1 330 70
PRINTER HP DESKJET 500	690.00
PRINTER HP DESKJET 500C	942.10
ODLDR KIT CITIZEN SWIFT 200	81 10
PRINTER EPSDN LX-400	315 40
PRINETR EPSON LQ-100	432.90
PRINTER EPSON LQ-570	680 10
PRINTER EPSON LQ-1070	946 30
PRINTER EPSON LQ-1170	1 163 10
PRINTER EP DN LQ-2550	1.991 10
DATA AUTO SWITCH 4-1 (MP-401B)	97 90
DATA AUTO SWITCH 8-1 (MP-801B)	149 90
DATA AUTO SWITCH 4-2 (MP-402)	144 80
CATA AUTO SWITCH BUFFER 4-2. 1MB	300 00

PLOTTER	DEM
PLOTTER ROLAND DXY 1100 A3	1.523 00
PLOTTER ROLAND DXY 1200 A3	1 896 40
ROLAND STIKA	1 162.20

MOUSE	DEM
MOUSE GENIUS ONE (GM D320)	26 10
MOUSE GENIUS TOD (6PLUS)	41 40
MOUSE GENIUS HI (GM F 302)	64 30
MOUSE GENIUS HI CORDLESS	94 30
GENIUS HI POINT	88 50

DIGITIZER	DEM
TABLET GENIUS GT-906,9×6	282 50
TABLET GENIUS GT-1212B 12×12	488 10
TABLET GENIUS GT-1812D 18×12	840 00

SCANNER	DEM
SCANNER HANDY GENISCAN GS-4500	181 90
SCANNER HANDY GENISCAN GS-B105A	276 40
SCANNER HANDY GENISCAN GS-C105 COLOR	742.90
SCANNER A4 HANDY W/PAP FEEDER	796 40

MODEM	DEM
MODEM 2400 INT (MNP5)	98 60
MODEM 2400 EXT (MNP5)	126 20
MODEM 9600 EXT (MNP5)	268 10
MODEM 2400 POCKET	116 10

PORTABLE PC	DEM
PC NOTEBOOK AT 386SX VGA 2MB/80MB	1 734.90

UPS	DEM
UPS 1000VA	726 00
UPS 1000VA DN-LINE	1 463 70

MULTIMEDIA	DEM
SOUND BLASTER LASER WAVE CLASIC	112 40
SOUND BLASTER LASER WAVE SUPRA 16	317 40
CD ROM DRIVE PANASONIC	455 70

DIFF	DEM
DISK BOX 5×5,25"	1 90
DISK BOX 10×5,25"	3.20
DISK BOX 50×5,25"	10 70
DISK BOX 100×5,25"	13 70
DISK BOX 5×3,5"	2.50
DISK BOX 10×3,5"	3.00
DISK BOX 100×3,5"	14 70
COPY HOLDER (411)	12.80
MONITOR DUST COVER	4.80
KEYBOARD DUST COVER	2.50
PRINTER STAND (355)	14 90
CPU HOLDER (396)	32.80
KEYBOARD DRAWER (431)	35.70
CLEANING DISKETTE FOR 5.25" (562)	7 10
CLEANING DISKETTE FOR 3.5" (563)	7 00
CLEANING MOUSE (564)	11 90
COMPUTER CLEANING KITS (565)	21 30
ANTI-STATIC MATS (585)	12 90
COMPUTER TOOL KITS (CKT 2019)	65 50
FAX MODEM CARD MNP5 V42BIS	126.20
FAX MODEM POCKET	138 00

CALCULATOR	DEM
PRINTING CALCULATOR PT-212 + AC ADAP	57 70
DESK ELECTRONIC CALCULATOR DS-1200B	29 00

ALARM	DEM
Kompletna ponuda alarmnih uredaja.	

UNIVERZITETSKI RADIO

Index

88,9 MHz FM Stereo

Nedeljom od 13 do 15 h (do 1. 09.)
emisija o računarima

"PC PLUS"

Autori emisije: Zoran Đukanović i Nenad Mitovski

Univerzitetski radio "INDEX"
Hilandorska 2, Beograd
Tel. 346-801, lokal: 196
Telefon u studiju: 330-060

"PC PLUS"
Preduzeće za računarski inženjering
Budimska 10a, Beograd
Tel./Fax: 337-120, 340-227

PC386,486 VESA-LOCALBUS-32bit !!!

SC 3040 DX ISA	SC4033DX LOCALBUS	SC4050DX LOCALBUS
AMD 386DX-40MHz RAM 4Mb / 128Kb cache FLOPPY 1.2 Mb HD 105 Mb ALPS 16bit IDE CONTROLER 16bit SVGA 1 Mb CIRRUS LOGIC - - WINDOWS AKCELER. 1280x1024 14" SVGA MONO MONITOR MINI TOWER, TASTATURA 101	INTEL 486DX-33MHzVESA RAM 4Mb / 256Kb cache FLOPPY 1.2 Mb + 1.44 Mb HD 250 Mb TEAC,WD,CONNER 32bit VL BUS IDE CONTROLER 32bit VL BUS SVGA 1Mb CIRRUS - - WINDOWS AKCELER. 1280x1024 14" SVGA COLOR MONITOR MIDI TOWER, TASTATURA 101	INTEL 486DX-50MHzVESA RAM 4Mb / 256Kb cache FLOPPY 1.2 Mb + 1.44 Mb HD 250 Mb TEAC,WD,CONNER 32bit VL BUS IDE CONTROLER 32bit VL BUS SVGA 1Mb CIRRUS - - WINDOWS AKCELER. 1280x1024 14" SVGA COLOR MONITOR MIDI TOWER, TASTATURA 101
1.850 DEM	**3.550 DEM**	**3.900 DEM**

386SX-33MHz, RAM2Mb, HD105Mb, FD1.2Mb, 14"MONOSVGA, 1Mb-WIN.AKC.....1550DEM

	CENE DELOVA	1-10 KOM.	11-30 KOM.
DISKOVI	HARD DISK 105Mb ALPS 13 ms	400 DEM	350 DEM
	HARD DISK 170Mb CONNER 17 ms	500 DEM	450 DEM
	HARD DISK 250Mb CONNER,MAXTOR,WD,TEAC 12-16 ms	700 DEM	600 DEM
	HARD DISK 540Mb CONNER 9 ms IDE III SCSI	1850 DEM	1700 DEM
	FLOPPY 1.2Mb / FLOPPY 1,44Mb	140 / 120 DEM	130 / 110 DEM
KARTICE	16 bit 2HD, 2FD CONTROLER + 2RS232 + 1 PARALEL	50 DEM	40 DEM
	32 bit VESA LOCAL BUS 2HD, 2FD CONTROLER / SA 4Mb CACHE	150 / 1150 DEM	130 / 1050 DEM
	SCSI ISA 2HD, 2FD CONTROLER 4Mb CACHE	1350 DEM	1250 DEM
	SCSI VESA, EISA 2HD, 2FD CONTROLER 4Mb CACHE	1450 DEM	1300 DEM
	ETHERNET 16 bit HE 2000	170 DEM	150 DEM
	FAX - MODEM KARTICA MNP5, V.42 BIS	200 DEM	180 DEM
SVGA	MONO MONITOR 14" 1024x768	250 DEM	220 DEM
	COLOR MONITOR 14" 1024x768	650 DEM	600 DEM
	COLOR MOHITOR 15" 1280x1024 RAVAH EKRAH HON INT.	980 DEM	800 DEM
	COLOR MOHITOR 17" 1280x1024 RAVAN EKRAN N.I. LOW RAD.	2200 DEM	1980 DEM
	CIRRUS LOGIC 1Mb, 1280x1024 WIHDOWS AKCELERATOR	220 DEM	190 DEM
	32 bit VESA L. BUS CIRRUS LOGIC 1Mb, 1280x1024 WIH AKCEL.	280 DEM	250 DEM
	32 bit VESA L. BUS S3 1Mb, 1280x1024 WINDOWS AKCELERATOR	400 DEM	360 DEM
PLOČE	386 SX-33MHz	220 DEM	200 DEM
	386 DX-40MHz-128Kb CACHE	380 DEM	300 DEM
	VESA L. BUS 486 DX-33MHz-256Kb CACHE	1000 DEM	950 DEM
	VESA L. BUS 486 DX-50MHz-256Kb CACHE	1350 DEM	1250 DEM
	VESA L. BUS 486 DX2-66MHz-256Kb CACHE	1550 DEM	1450 DEM
PRINTERI	LASER HP IV-2Mb 600x600 / RAM 4 Mb	3600 / 800 DEM	pozovite
	LASER HP IVL - 1 Mb 300x300	1900 DEM	pozovite
	EPSOH A4, LQ 570-24 PIN, KABL (YU SET 50 DEM)	900 DEM	850 DEM
	EPSOH A3, FX 1050-9 PIH, KABL, YU SET	1150 DEM	1050 DEM
	EPSOH A3, LQ 1070-24 PIH (YU SET 50 DEM)	1250 DEM	1150 DEM
	PAHASOHIC A4, 1121I-24 PIH, KABL IIIII	600 DEM	550 DEM
RAZNO	MEMORIJE SIMM 1Mb / SIMM 4Mb	130 / 550 DEM	pozovite
	MATEMATIČKI KOPROCESOR IIT 387DX-40MHz	200 DEM	180 DEM
	MINI TOWER + 200W / MIDI TOWER + 200W	160 / 200 DEM	140 / 180 DEM
	PROFESIOHALHA TASTATURA 101 / ALPS SA KLIKOM	70 / 120 DEM	60 / 110 DEM
	MIŠ - MICROSOFT KOMPATIBILAN, SA PODMETAČEM	50 DEM	40 DEM
	STREAMER 250Mb	690 DEM	600 DEM

ISPORUKA ODMAH **GARANCIJA 12 MESECI**

ŽIRO RAČUN: 60815-601-64787 RADNO VREME 9h-17h

TEL: 011/332-607
FAX: 011/345-126

PREDUZEĆE
ŠUTLIĆ
&
MIKRO DIZAJN

11000 BEOGRAD
Kosovska 32, I sprat
SEPTEMBAR 93.

PC ONOMATOPEJA

Kada je atvarao PC, Bog nije brinuo puno o tome de ge obdeni zvukom. Razvojem ovog računera, međutim, počela je da se javlja potreba za boljim zvukom, pe su se pojavile mnoga zvučne kertice I programi koji varaju zvučnik. Ovaj tekst ne predstavlja test dodataka za PC, već uopšteni pregled genarisanja zvuke ne njemu, a nepiseo ge je autor čije se ime vezuje za jeden izuzetno ambiciozni domaći projekat primene PC zvuka u radio-difuziji, *Redio Pingvin SofSound*.

Aleksandar Šušnjar

U PC računarima na postoji posebno zvučno kolo koje bi genilo zvučnik. IBM je iskonsko slobodan ka nal se tajmera 8253 (kasnije u AT u 8254) za genen sanje zvuka pravougaonog bilasa određene frekvencija. Ovaj čip inače služi kao delilelj frekvencija 1193180 Hz što se dobija daljenjem sistemskog kloka (prvog PC-je) od 4772720 Hz se 4. Na taj način ja PC bio osuđan na aviranje četvrtka kao proizvoda de ljenja tajmerskog kloka se nekom 16-bitnom vrednoš ću. Ovako „moćan" zvuk je u početku konšćen za sig nelizaciju greške i sličnih poruka a kasnije su se po javili muzički programi koji su sviral čitava numera Čak je i sam IBM napravio jedan takav program stari MUSIC pisan u bejziku

Uporedo s pojavom korisničkih i poslovnih progra me pojavljivale su se i igra od kojih su se prve zado voljavale tišinom ili pištanjem i bipranjem. Kasnije su se mogle naći i melodije kojima bi bio dovoljan jedan kanal (jedna nota u jedno vreme) ali takvih n je mnogo. Tada je polako počela revolucija slična onoj koje se odigrala na starim dobrim ZX Spectrum „sistemi ma". Pojavili su se razni programi koji su izvlačil „ne moguće" iz PC-ja, kao npr. *Digger* i *VMusic*. Način na koji su oni svirali svoje melodije su meltene (možda i bukvalno) preneseni sa gorepomanutog ZX Spectru ma. Sve se svodilo na pokušaja da se odsviruju 2 do 3 note istovremeno dok je cilra od 4 kanala delovala nedostižno. Recimo da je filozofija njihovog rada bila kompletno pogrešna. I da se u startu iz PC je mogla izvući mnogo više ali se jednostavno loga niko nije selio. Negde u to vreme se pojavila

JEDNA OD PRVIH

U druge računare su u to vreme bili ugrađivani čipovi koji su generisali zvuke raznih boja na čitava 3 kanala uz dodatni šum. Takvi su se mogli sreti na računarima C64 Amstrad CPC seriji Lola 8 ili ZX Spectrum 128 itd. Uglavnom je korišćen AY 3-8912. Neko kome je dosadilo PC jeftine pištanje i ko se seto pomenutih računara pokrenuo je shvatio i tako je nasta la *AdLib Sound Card*. Radilo se o prostom FM generatoru koji je mogao na 11 kanala da pušta tonove raznih frekvencija i boja odnosno šumova. De budu mo precizniji, 6 melodijskih i 5 šumova. Potrebno je bilo poslati kartici sve potrebne parametra (frekvenciju envelopu osnovni oblik) za neki kanal i svirka bi počela. Ovakav zvuk je umnogome bio sličan zvuku pomenutih 8-bitnika uz tu razliku da se *AdLib*-om mogao generisati zvuk na više kanala. *AdLib* filozofija savršeno odgovara večini igara jer se bez većeg Ire šenja dragocenog procesorskog vremena mogu svirati lepe melodije i pravi interesantni efekti. Pod nekim na prstarano čestim interaptom (recimo za takve stva n predvidenim 1Ch nominalno 18.206481 Hz) može se napisati rutina koja će samo kada je to potrebno, promeniti parametre „svirke" na nekim kanalima. Na taj način su se bez gotove likakvog gubitka procesor skog vremena mogle svirati jako lepe muzičke nume ra i povremeno aktivirati neki efekti dok se glavni pro gram uglavnom bavio – igrom. Takođe datoteke koje sadrže melodiju su jako kratke u klasi dužina MIDI datoteka.

Jedino ljudski ijtas i muzički snimci se nisu lepo mogli rešiti preko FM sintetizatora ali to i nije bilo to liko neophodno. I pored mnogo boljeg zvuka od standardnog ove kartice u neku ruku nije imale sreće. U to vreme ja vladala lobija u PC svetu od svega ne standardnog. *AdLib Sound Card* je bio okarakteri san kao takav. Bilo je špasno zvati BIOS interapt rutine a kamoli direktno „razgovarati" se hardverom. Ni BIOS ni DOS nisu znali (niti znaju) za *AdLib*. Micro soft i IBM nisu predvideli posebne funkcije za sviranje a do sade još nije usvojen drajver za tako nešto ako

Izuzmemo *Windowse* (ipak čak i u njima nije sve re šeno!)

Zbog svega toga se pojavila tek poneka igra koja je umela da svira (i na ovoj kerti. Zbog te lobije su se i dalje razvijali programi koj su direktno mučili PC zvučnik jer je i to bio manji greh od sviranja na *AdLib*-u. Ipak, polako je *AdLib Sound Card* postajao stan dard. Nije ga bilo bas toliko teško koristiti a progra meri su se polako osiobadali lobije od direktnog „prčkanja" po hardveru. Veliki problem je bio i u tome što većina igara nije posebno stavljala do znanje da „ume i lepo da svira". Možda se to mogio I pročitati na originalnim pakovanjima, ali je tada još uvek nije bilo dovoljno zabrenjeno kopirati softver pa su se ve ćinom dobijala piratske kopije

DIGITALNO-ANALOGNA...

Drugi način da se proizvede zvuk pomoću ra čunara je upotreba digitalno-analognih konvertora. Ovo je pored svih mana većini najinteresantniji način ge nerisanja zvuka pa ću mu posvetiti najviše pažnje

D/A konvertori su elektronska kola koja pretvaraju stanje na digitalnim ulazima (praktično b narni broj) u neki napon iz zadetog opsega na izlazu. Razlikuju se po broju binarnih cifara (bita) koja koriste, brzini rada, dozvoljenom opsegu izlaznog napone načinu konver zije i kvalitetu. Postoje gotovo sve moguće komb nacije pomenutih parametara. Recimo de su najinteresan broj 8 10 12 i 16 bitni brzine (frekvencije uzorkovanja) od 10 kHz (mada ih ima i od 4 kHz!) bipolarni (naponski opseg može ići od negativne do pozitivne vrednosti simetrično u odnosu na nulu III ne)

Načini konverzije su različiti. Od najpanih običnih težinskih (otpornih) mreža kao što je to kod Covox a (ako se ti otpornike može nazvati mrežom) do direk tnih koji „jednostavno" imaju n otpornika i neki digital no kontrolisani analogni multiplekser koji bira sa koje će tačke proslediti napon. Ovi poslednji su najkvalitet niji najskupiji najbrži uglavnom najtačniji ali m je mane što zahtevaju relativno veliku broj komponenti za mali broj bita. Najčešće se koriste u video tehnici – 8-bitni D/A konverton za sveku R. G. B komponentu pojedinačno. Ima i konvertora koji rade po principu šinnske modulacije. Jako velikom brzinom se emituju četvrtasti impulsi čija širina (trajanje) odgovara zada toj vrednosti. Zatim se takav signal prosleđuje kroz integrator i tako dobijemo relativno konstantan napon na izlazu. Zvuk se generiše tako što se veliki broj puta u sekundi pošalje preva vrednost D/A konverteru. Na taj način sa mogu formirati razni talasi uz određenu grešku koju sledi iz (ne)tačnosti i učestalosti konverzi je

Ovakva filozofije se primenjuja u (novim) progra mima koji puštaju sve i svašta kroz običan PC zvuč nik. Pri tom se tajmer koristi kao generator četvrtki e elektronike od lajmera do ti samog) zvučn ke kao integrator. Na ovaj način se može izvući tačnost od 5-6 bita zavisno od željene frekvencije uzorkovanja ako ne praterujemo. U vazi se ovim postoji jedan prilični problem. Ovakav način genarisanja zvuke je jako osetljiv na vremenske greške. Te greška u principu na postoja po pitanju šinna impulsa već po pitanju njegovog početka – širine nule. Pošto se kontrola mo guća vršti softverski to se naprednije radi uz pomoć nekog interapta (mliče tako treba) ali ako pn tome želite da se paralelno izvršava i još nešto dolazi do silnih fluktuacija u početku izvršavanje interapt rutine koje mogu stvoriti nesnosna pištanja, krčanja, zujanja pa i „zuborenje" i „grgoljenje"

Većina prostih težinskih (otpornih) mreža se čak i ne mora svrstati u D/A konvertore. Jako su nelinear ne i nepreciznе i mogu se konstrti samo u amaterskim. Ovakva kola se jako mogu napraviti i u sopstve noj izradi e neke šeme se mogu naći uz razne „svira če" muzičkih modula i samplova. Učestalost konverzi je (frekvencija uzorkovanja) direkino utiče na frekvent ni opseg koji se može „odsvirati". Kaže se da je mak simalna frekvencija koja se može reprodukovati jed naka polovini frekvencije uzorkovanja. To jeste princ pijelno i matematički tačno, ali se pri vršim frekvencija ma javlje sve veća greška u odnosu ne originai pe se lime gubi i boja. da ne pričamo o frekventnom od zivu u toku snimanja

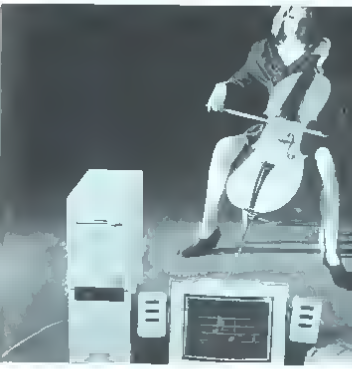
Uz pomoć D/A konvertora možemo genenisati ja ka složene zvučne talase praktično sve koje bi ljud sko uho razlikovalo. Samo genansanje zvuka može bih preko nekog matematičkog modela dakle sintet zovano ili možemo reprodukovati neki snimak. U pr vom slučaju softver može npr. matematički generisati i sabirati (ekvivalent miksovanju) sinusne trougaone pravougaone i druge slične funkcije raznih perioda (frekvencija) i na taj način formirati muziku. Ovako bi radile neka vrsta simulatora FM sintetizatora.

ANALOGNO-DIGITALNA KONVERZIJA

Drugi način je mnogo interesantniji. Kao što po stoje D/A tako imamo i A/D (analogno-digitalnaj konvertore koji rade obrnutu stvar. Napon iz nekog opsega pretvaraju u neku binarnu vrednost predstavljenu na digitalnim izlazima. Kod A/D konvertora priča o ne du je mnogo komplikovanija od one za D/A pa ću se zadržati samo na osnovnim stvarima. Sama A/D kon verzija je mnogo kompleksnija od D/A. Interesantan je metod sukcesivne aproksimacije gde se koristi je dan D/A konvertor analogni komparator i kontrolna logika. A/D konvertori polazi od neke vrednosti (pogađa) pošalje je D/A konvertoru i zatim analognim komparatorom poredi mereni napon sa naponom na izlazu iz D/A konvertore. Zavisno od rezultata vrši ko rekciju vrednosti i tako iterativno dok na „pogodi" od nosno na dođe do vrednosti čije je greška manja od zahtevana tačnost merenje. Ceo ovaj proces je dosta brz ali je u svakom slučaju vremenski mnogo kritičniji od D/A konverzije. Dalje A/D konverton ovog tipa s lošijeg kvalireta se leže snalaze se promenljivim na ponskim veličinama ne ulazu koja uma da im pobegne. U tom se slučaju dobija neka nedefinisana vred nost

Imao sam prilika da radim sa A/D konvertorom koji dozvoljava nešto više od 4000 merenja u sekundi na nekih 7 tačno izmeranih od ukupno 12 bita. Moram sam da uzorkujem zvuk se kasele koji sadrži i više frekvencija pa se pomenuti problem često javljed Sreća u nesreći je bila da se greška uvek manifestovala kao neka iz malog skupa veličine (uglavnom krajnjih) pe sam mogao da napišem program koji kada naiđe na takvu veličinu koje odskače od onih oko nje vrši zamenu interpolisanom vrednošću. Takođe kasetu sem pušiao duplo sporije nego pn snimaoju taka da je efektivna učestanost uzorkovanja bila oko 8000 merenje u sekundi što je bilo podnošljivo

Što se liče frekventnog opsega zvuka koji se mo že snimti važi da je MAKSIMALNA frekvencija take da jednaka polovini frekvencije uzorkovanja. Stvari se međutim komplikuju. Recimo da uzorkujemo na 10 kHz a da pokušavamo da snlimimo najobičniju „četvrtku" od 5 kHz. Ako se laze ove dve frekvencije pe klope formula važi. Šta se dešava ako se trenuci me renja poklope se tvcama četvrtke. Ili nuiema sinusoida. Dobijamo ili šumove u prvom slučaju ili jako lepo snimljenu tišinu u drugom. A tek ako frekvencije nisu idealne i nisu konstantne kroz vreme može se dobiti



SB 16 MIDI Player	
Verzija	apsolutna
Broj instrumenata	256
Preduhi	82
Udaraljki	66
Razlih oblika	66
Grupa udaraljki	42
Rekonfigurisanih	0
Efekata	50
Downloadable	192
Broj kanala	32
Multitimbral	da
MPC Level 1	da
Tehnika sinteze	E-mu
Emulira MT-32	da
Procesor	68000^10
Prilozeni softver	CakeWalk WSFannel WBPlayer WBMode

Jedina kartica SoundBlaster 16 ima ugraden MIDI Player sa karakteristikama kao u tabeli

snimljeni talas „prave" frekvencija sa oscilujućom amplitudom!

Pomenuo sam kako utiče frekvencija uzorkovanja na finalni snimak ili reprodukciju. Jako važan faktor je i broj bita – preciznost upotrebljenih konvertora. Sa većim brojem bita imamo manju grešku u merenju i samim tim manje šumova. Takođe, broj bita utiče na boju i mogućnost kompleksnosti zvuka koji se snima. Imao sam prilike da čujem snimke koji su mogli neka ko da se reprodukuju pomoću 8 ili nikako (baš bu kvalno nikako – krš, šušti, zuji samo ne svira) ne 8 bita. 6 bite jesta malo, ali je dobar primer.

Previše su se i zvučne kartice (dodaci) koji su koristili D/A (i ponekad i A/D) konvertore za generisanje zvuka. U početku su to uglavnom bili dodaci koji su imali samo taj način generisanja zvuka. Najbolji primer je COVOX dodatak koji se priključuje na paralelni port. Ove jako jeftina kutijica ima otpornu „mrežu" od 8 otpornika i ume da svira čak i kada vi želite samo nešto da štampate. Izuzetno je jeftina, može se napraviti i u kućnoj izvedbi, može da se priključi na gotovo sve računare (ne računeući softver) ali ima najgori kvalitet zvuka prvo zbog načina konverzije a drugo zbog direktne interferencije digitalnih šumova, pa se često čuje „zubor vode" kada se vrši neki transfer podataka ili „grgoljenje" kada procesor upadne u neku dužu petlju. To je još jedan neprijatelj u ovom slučaju svih vrsta zvučnih kartica. Problem je rešiv ali je potrebno projektovati i štampanu pločicu ne poseban način dodati nešto više elemenata, a ze profesionalni kvalitet galvanski odvojiti računar od konvertere pomoću recimo optokaplera.

NEVOLJE SA MEMORIJOM

Generisanje zvuka uz pomoć D/A konvertora je interesantno sa gledišta raznovrsnosti mogućih zvukova i samog kvaliteta zvuka. Dalje ne postoji neki naročit problem vezan za broj kanala jedan 8-bitni D/A konvertor normalne brzine može uraditi mnogo više nego bilo koji FM sintetizator i on tom ga simulirati ako imate dovoljno brzu baznu mašinu. Preko jednog takvog kanala možete svirati desetine pa i stotine nota bez problema. Uglavnom jedino pitanje koje se postavlja je da li se radi o stereo ili mono kartici. Za mono zvuk vam je potreban po mekar jedan A/D (osim ako ne želite da snimate) i D/A konvertor. Što se tiče stereo zvuka potrebno su vam dva D/A (ili jedan dupli)) i dva A/D konvertore (ili jedan...ali dupio brži tako da može da stigne da izmeri napon na oba kanala u isto vreme). Pitanje većeg broja kanala od 2 se postavlja jedino u profesionalnim krugovima gde vam PC Atari Amiga ili bilo koja druga igračka ne mogu pomoći – tu se koriste jače mašine. Inače par celobrojnih sabiranja više-manje ne predstavlja problem za glavni procesor.

Glavna mana ovog načina generisanja zvuka je u količini potrebne memorije. Za primer mogu poslužiti običan *Sound Sculpter* u igrama sa oko 13500 bajtova u sekundi do 240 000 bajtova u sekundi za profesionalne digitalne sisteme. Ako snimke (semplove) držimo u RAM-u verovatno će to biti više jako kratki ili jedan-dve sredine dugačka sample maksimalno. Moguće je povremeno učitavanje semplove se hard diska, ali šta ako je vreme učitavanja kritično ili je, pak, sampl bako dugačak da ne može da stane u raspoloživ RAM. Ovakav problem se javio u Radio Pingvinu gde su semplovi dugački po više desetine megabajtal

Iz ovakvih razloga u početku AD/DA dodaci nisu bili naročito popularni ni među programerima (zbog potrebne brzine rada da se lolika količina podataka prenese a da ostane vremena i da se radi još nešto) ni korsnicima (zbog broja utrošenih disketa ili dragocenog mesta na ogromnom hard disku od čak 30 Mb!) Tek se u novije vreme pojavljuju programi koji koriste D/A konvertore do maksimuma, ne računajući softver koji se dobija uz karticu. Problem sa velikim gutanjem memorije je delimično rešiv na više načina.

1) Kompresijom sampla nekim od poznatih algoritama. Najčešće (i najbolje) se to radi zapisivanjem razlika između izmerenih vrednosti na marjem broju bitova od onog kojim se inače snima. Ako se razlika ne može zapisati u tom broju bita upisuje se najbliža moguća ili se uvodi kod koji označava da sledi prepisana vrednost iz sampla. Postoje i bolji metodi kojnisam u mogućnosti da izložim široj javnosti zbog Implementacije takvog algoritma u *Pingvin SoftSound* sistem gde je prosečna kompresija zvuka 3 1 u odnosu na originalni sample dez ikakvog (pa i teorijskog) gubitka na kvalitetu. Uz neki dozvoljeni gubitak kompresije bi lela i do 8 1. Šteta je što neki (ovakev zašto ne?) algoritam nije primenjen u izradi CD-ova – zamislite CD od 3 do 6 sati e pritom normalnog prečnika!? Inače *Pingvin SoftSound* sistem je baziran na profesionalnom CD kvalitetu što podrazumeva stereo, 16 bita uzorkovanje 44 1 kHz po kanalu i jako mali nivo šuma. Inače DSP (Digital Signal Processor) može leko da se iskoristi ze ovu svrhu ako ga me. Kompresija je pri čno univerzalan metod za štednju memorije i upotrebljiva je u svim oblastima računarstve. Ipak, ako je nešto stvarno veliko, onda i kada je kompresovano može biti preveliko!

2) Organizovanje podataka kao izvestan broj semplova koji se mogu puštati različitim brzinama odredenim redosledom i jačinom paralelno sa drugim. Ovaj metod koriste muzički moduli preneseni se Amige. Mogu deveti jako lepa zvučne slike ali sa nekim kraju ipak dobije jako neprofesionalan kvalitet zvuka u svim pogledima. Idealan je za demoe. Inlice poza dinsku muziku i igre (kako bi Inače dospeo baš as Amige?) ali nikako nije za ljude od muzike. Ovakve organizacija na omogućave razne prelaže elekte kao što su efekti raznih prostorija razonancija samih muzičkih instrumahata e ni instrumenti nisu prisetano realni za više od igre (nadite dobar klavir ili violinu)!

3) Prepuštanje pamćenja dužih semplove dovoji ne velikom i brzom hard disku. Za ovo je neophodna i mogućnost direktnog ili brzog slanja podataka ličnih sa hard diska ka zvučnoj kartici – ovo je još jedan metod koji sa koristi u *Pingvin SoftSound* sistemu. Da bi se ovo izvelo koristi se metoda višestrukog bafe-rovanja. Za najprostije primene reći se sa dva bafera jedan se koristi za učitavanje sa diska zatim se u potrebnim vremenskim intervalima i količinama prenosi u drugi odakle se šalje zvučnoj kartici. Velika pomoć ovome su veliki FIFO baferi na samoj zvučnoj kartici, koji zamenjuju poslednji bafer drugi u ovom slučaju. Naravno samo ako postoje. Pušlanje semplove direktno sa hard diska je pogodno samo za uzak krug korisnika. Što se komercijalnog softvera tiče nikada nećete biti sigurni da će računar na kome će se program izvršavati imati dovoljno brz hard disk i prenos se njega. Naprotiv stvari je idealna za potrebe kao što su npr. u Radio Pingvinu zbog kojih je i napravljen *Pingvin SoftSound* sistem e to znači puštanje džinglova, reklama, upadica čitavih muzičkih numera, nadsinhronizacija uz pomoć računara itd.

Kao zaključak o upotreba A/D i D/A konvertora ne zvučnom polju pomenuću da su 8-bitna kartice dovoljne za većinu potreba. 16-bitne su neophodne samo na mestima gde se zahteve zvuk punog kvaliteta. U svetu se koristi i 20-bitni zvuk ali uglavnom samo u toku masteringa zbog lakše manipulacije sa više kanala. Na kraju se opet sve prebaci na 16 bita. Što se tiče frekvencije uzorkovanje dovoljno vam je 4kHz za (naš) telefonski kvalitet od 10 kHz za relativno normalan zvuk od 20kHz za dobar i iznad 40kHz za „onaj pravi". Zgodno je da kartica podržava uzorkovanje na frekvencij od 44 1 kHz (makar samo pri reprodukciji D/A) i duplo i četvorostruko manju frekvenciju 22050 Hz i 11025 Hz koje inače nameće i *Windows* kao MPC Level 1 standard.

MIDI

Postojale su i karte sasvim druge filozofije. Sa njime PC nije mogao da sviia već da kontroliše druge elektronske muzičke instrumente preko tzv. MIDI porta koji predstavlje jednu vrstu serijskog interface-a. Uz pomoć ovakvih kartica vaš PC može da komanduje (e i da prima komande) sa više instrumenata istovremeno slanjem odnosno primanjem informacija o notama koje treba odsvirati zajedno sa pratećim efektima. Na taj način se računar može koristiti kao jako dobar (najbolji) ekvenosi koji u startu može služiti kao običan asistent koji pamti šta ste odsvirali recimo pratnju a zatim lo šalje vašoj klevijatun koje će tada svirati i pratnju (sa računera) i ono što vi budete svirali. Tada računar na pratnju dodaje i ono što ste tada odsvirali itd. Tako možete izgravah ceo jedan mali orkestar. Eventualne greške se lako čaju ispraviti u ugrađenim editorima e može se čak kongovati i izražajnost. Takođe lako ćete probati kako zvuči ista stvar na nekom drugom instrumentu. Vaš PC vam tako može služiti kao jako izuzetno pomagalo pri komponovanju koje uz to može i da svira na 5-6 instrumenata u isto vreme. dok pri tom sam nije ni jedno „dirp" izustio. Uglavnom je moguća kontrola od 16 do 32 kanala/instrumenta – bez ili sa „multitimbral" moda koji omogućave sviranje više nota po jednom kanalu. Od čisto MIDI kartica pomenuću Rolandova LAPC 1 i SCC-1 koje su postale svojevrstan standard u ovoj grupi.

Ovakve filozofija odgovara profesionalcima koji mogu iskoristiti PC je kao sekvencer za svoje klavijature. Za druge oblasti računarstva MIDI nije odgovarao zbog toga što nikada niste mogli biti sigurni šta je i kako je organizovano „ono" se druge strane MIDI kable. Tek se se pojavom jeftinijih MIDI box-ova pojavilo nakoliko igara koje su prilagođena strogo jednom box u kao što je to npr Rolandov MT-32 ili CM-64. MIDI box je uređaj koji uma da svira ono što mu stižu se MIDI porta, ali nema (ali klavijatura da bi bio kompletan sintisajzer. Cela kontrole se vrši preko MIDIja a na sebi uglavnom nema mnogo toga.

MIDI najčešće koriste profesionalci koji danas mogu neći jako dobre programe i za PC-ja kao što su npr. *CuBase PC CakeWalk Apprentice* i slični. Kod nas je medium verovatno zbog cene najrasprostranjeniji Atari ST sa *CuBase*-om. Ako želite da se bavite muzikom pogotovo komponovanjem MIDI vam je neophodan.

HIBRIDI

Trebalo je spojiti dobre osobine kartica prvog drugog i trećeg tipa. Pojavljivale su se razne kartice koje su ilmale više ili manje mana i nikada nisu bile ono prave. Jedan od prvih pokušaja na tom polju je

Uporedni pregled zvučnih kartica

Kartica	AdLib Sound Card	Covox Speech Thing	Covox Sound Master	Sound Blaster	Sound Blaster 2.0	Sound Blaster Pro	Sound Blaster 16	Sound Galaxy NX	Pro 16 Audio	Pingvin SoftSound
AdLib kompatibilnost	da	ne	ne	da	da	da	da	da	da	ne
SB kompatibilnost	ne	ne	ne	nemvne	da	da	da	da		ne
FM kanala	11	nijedan	4	11	11	2-op 20 4-op 11	2-op 20 4-op: 11	11		–
Melodijskih	6	nijedan	3	6	6	2-op. 18 4-op 6	2-op 16 4-op: 5	6		–
Šumnih	5	nijedan	1	5	5	5	5	5		–
Linijski ulaz	ne	ne	ne	da	da	da	da	da		da
kroz mixer	–	–	–	ne	ne	da	da	ne		da
šumovi	–	–	–	postoje	mali	jako mali	minimalni	mali		nema
gain control	–	–	–	ne	ne	da	da	ne		da
auto gain control	–	–	–	da	da	da	da	ne		ne
Linijski izlaz	da	ne	ne	da	da	da	da	da		da
master volume	da	–	–	da	da	da	da	da		da
šumovi	?	–	–	postoje	mali	jako mali	minimalni			nema
Mikrofonski ulaz	ne	ne	ne	da	da	da	da	da	da	ne
kroz mixer	–	–	–	ne	ne	ne	da	da	da	–
šumovi	–	–	–	značajni	mali	jako mali	minimalni			–
gain control	–	–	–	ne	ne	da	da			–
auto gain control	–	–	–	da	da	da	da			–
linija na stražnjoj strani	?	Behm	Behm	4W	4ohm	4W	4ohm	4W	4ohm	4W
A/D konverter	ne	ne	ne	mono	nemo	stereo	ali multiplex	stereo	mono	
rezolucija (bita)	–	–	–	8	8	8	8 / 16	8		16
brzina	–	–	–	13.6 kHz	15.15 kHz ?	1x45 kHz	2x22.5 kHz	2x45 kHz	44.1 kHz	
linearnost	–		–	dobra	dobra	vrlo dobra	odlična			odlična
digitalni šum	–	–	–	mali	jako mali	minimalan	nema			nema
16-bit (AT) ISA, ne	nije kartica	ne	ne	ne	da	da			jedino	
D/A konverter	ne	nema	nema	mono	nemo	stereo	stereo	nema	stereo	stereo
rezolucija (bita)	–	8	8	8	8	8	8 / 16	8	8 / 16	16
brzina	–	?	lepod 20 kHz	4-23 kHz	44.1 kHz	1x45 kHz	2x22.5 kHz	2x45 kHz	44.1 kHz	44.1 kHz
linearnost	–	odvratna	zadovoljava	dobra	dobra	odlična	perfektna	dobra	perfektna	perfektna
digitalni šum	–	ogromna	velika	ok	ok	mali	minimalan	mali	minimalan	nema
filter		na	loš	ostavljiv	odličan	odličan	perfektan	dober	perfektan	perfektan
16-bit (AT) ISA	ne	nije kartica	ne	ne	ne	da	da		da	jedino
DSP	nema	nema	nema	prost	prost	prost	odličan*			nema
kompresija	–	–	–	da	da	da	da	da	da	–
run-time ekstenzije	–	–	–	ne	ne	ne	da	ne	ne	–
programabilan	–	–	–	ne	ne	ne	da	ne	ne	–
FIFO buffer		nema	nema	mali	mali	dober	solidan	mali	nekonfigurisano	veliki
Mikser	nema	nema	nema	nema	nema	dobar	odličan	nema		teman
ulaz na zvučnike	?	nema	ima	nema	nema	ima	ima			nema
linija na zvučnik	nema	nema	ima	nema	nema	ima	nema	nema	nema	nema
CD ROM kanal	nema	nema	nema	nema	nema	ima	ima	nema	ima	nema
MIDI kanal	nema	nema	nema	nema	nema	nema	ima	nema	nema	nema
dinjestik portovi	?	nema	paralelni	analogni	analogni	analogni	analogni	analogni	analogni	nema
MIDI gen	ne	ne	ne	opciona	opciona	opciona	opciona			nema
MIDI Player	nema	nema	nema	nema	nema	nema	intern**	nema	nema	nema
MIDI IR	ne	ne	ne	dopisla	dopisla	opelja	bila koji	dopisla	dopisla	ne
Cena MIDI kit-a	–		–	?	200	200	kog?	140		–
CDROM adapter	ne	ne	ne	ne	ne	da	da	da	da	ne
Priključno	?	ništa	Speech Master i sample player	audio kabl	audio kabl	audio kabl	audio kabl	mikrofon	audio kabl	
Softver	? ali ga gotovo svi podržavaju	nema pušač nemaju-ova ali ga mnogi moduli plejeri poznaju		FM organ CMFplay PlayMIDI sb-talker dr sbaltkr parrot VoxKit MMPlay Spellbox	FM organ CMFplay PlayMIDI sb-talker dr sbaltkr parrot VoxKit MMPlay	FM organ CMFplay PlayMIDI sb-talker dr sbaltkr parrot VoxKit Vedit Voyetra MMPlay Mixer	FM organ CMFplay PlayMIDI sb-talker dr sbaltkr SoundFX VoxKit WavKit MMPlay Mixer uz WB WBMods PlayMIDI			SoftSound DiejEsther WaveShow Recorder Player
Windows softver	uz Windows idu drajveri	nije bilo Windowsa	nije bilo Windowsa	nije bilo potrebno	Juke8ox	Juke8ox Mixer Scheduler	Juke8ox Mixer Monologue WaveStudio SoundE/LE Scheduler Mosaic uz WB CakeWalk WBPannel			nema
Prodavac	zastareo	zastareo	zastareo	kmisl (klonovi)	AdaCom	PIN Elektronik	CjemTrad	AdaCom	kmisl	Rasko Pingvin
Kontakt adresa	nepoznata	nepoznata	nepoznata	Bulevar Lenjina 166b 135-420	Pro Musica Čika Ljubina 12	629-233	Slavana Dukića 2	772-435	Genex sprinan 388	2222-852
Okvirna cena u DEM	<50 eko je nadete	<15 skoplja ga sami da s stpernika.	<25	originl 200-250	klonovi 100-150	390	SB Pro 450 +MIDI kit +Voyetra +Tetra = 700	SB16 390 SB16MIDI *** SB16ASP 850 WaveBlast. +450	300	90b

* Samo Sound Blaster 16 ASP. ** Samo sa dodatim Wave Blaster-om. *** Cena nije poznata nema ga u ponudi. Samo se zna da postoj

bio Covox Sound Master koji je objedinio AY-3-8912 čip sa D/A konvertorom sa dve džojstik porta za obič ne preklađačke džojstika koji bi se uz priložen softver ski simulator ponašali slično mnogo skupljim analog nim. Nije bio kompatibilan sa uz previše rasprostranjenim ali ipak standardom se *AdLib Sound Card* Zbog svega toga ove kartice nije uspela na tržištu

Uvideli kako su se drugi opekli u Creative Tec hnologies je nastale kartica po imenu *Sound Blaster* koji je uspelo da već duže vreme bude apsolutni standard uz zvučnom polju Usvojene je kompatibil nost sa *AdLib*-om koji je tak nekako smogao uteku popularnost dodal joj 8-bitni A/D i D/A konvertorski par i (skoro) MIDI port (potrebno je dokupiti MIDI kit) i naravno je bum ua tržištu Sve što je do tada pod ržavalo *AdLib* podržavalo je i *Sound Blaster* a on je lome još dodao I mogućnost snimanja i pušianje digi talnih sempiova Njegov A/D i D/A par nije bio bog zna šta Dozvoljavao je uzorkovanje do 13 5 kHz što je ipak, tada bilo apsolutno zadovoljavajuće MIDI je takođe bio u startu upotrebljiv sa postojećim softve rom a tu se pojavio i poseban sekvencer za *Sound Blaster – Voyetra* (lako ga neki hvale daleko mu lepa kuća)

Da ne bi bio samo spoj starih stvari u SoundBla ster je dodel i čip (reklamiran kao DSP) za hardvarsku kompresiju i dekompresiju zvuka na način koji sam ranije u tekstu objasnio uz opasku da postoji gubitak kvaliteta od neodređenog do drastičnog Nudile su se konstantne kompresije od 2 1 3 1 i 4 1 Pored toga dodel je I port za dva analogna džojstik porta koji se nalaze zajedno sa adjustirenim MIDI portom ua jed nom DB-15 priključku

Ova kartica je doživela toliku popularnost da su se pojavil mnogi kompatibilci i potpuni klonovi Kloneo vi su imali iste karakteristike ne papiru uglavno razli čit softver od originala bili su jaftiniji ali i lošije izrade, pa samim tim I kvaliteta od uzora. Kompatibilna kartice su sa zadovoljavale istim ili sličnim (poprilavljenim) karakteristikama kao što je na primer veća brzina cije uzorkovanje A/D ili D/A konvertora uz razne dru ge dodatke Iakođe manju cenu

Sound Blaster je doživeo nekoliko novih verzija u kojima su poprilavljena neka karakteristika originala frekvencije uzorkovanja broj FM kanala stereo zvuk 16-bitua rezolucija Novi Blasteu podržavaju MPC Le vel 1 standard (Multimedia PC Level 1 uvedeu u *Win dows-u*) koj dozvoljava uzorkovanje na fiksnim fre kvencijama od 11025 Hz 22050 Hz i 44100 Hz ne š li 16 bita stereo ili mono Varzlju 1 6 ne smatram uekom revolucijom tako de je ueću dalje ni pominjati *Sound Blaster* PRO uvodi stereo zvuk mogućnost uzorkovanje od 4 kHz do 45 kHz ukupno (max 22 5 kHz po kanalu za stereo) Takođe poboljšan je i FM deo tako da PRO nudi 20 kanala (15 melodijskih i 5 šumnih) ali mogu da primetim da se dodatni kauall retko (čitaj nikao) koriste

RAZNI DODACI

Sound Blaster kompatibilci su doprineli (morali su nešto da dodaju) zbrci koja postoji na tržištu Jasno je da se ovakve kartice uajčešće koriste za igre pa je donekle razumljive i potreba za uvođenjem analog nog džojstik porta koji važi kao IBM-ov standard na PC-ju

Međutim u međuvremenu se pojavid I CD-ROM sa svojim izuzetnim kapacitetom Pojavili su se i prvi SCSI (I SCSI 2) adaptau za PC ja koji su napravili opšti haos jer ulsu bili međusobno kompatibilni Adapteu su se uglavuo isporučivali uz sam uređaj (hard disk CD-ROM) koji ja mogao samo sa njim da radi lako to nije SCSI filozofija. Zamislite računo de želite SCSI hard disk i SCSI CD-ROM Bila su vam potreba ua dve adaptera koja bi bila vrlo predodređena da se posvađaju (i potuku) oko pitanja adresa, DMA kanala interapta I drajvera. Tu je nakome palo na pamet da CD-ROM uma I da svira što je dovelo do uvođenja nekog adapter sa/interfejsa za CD-ROM na samu zvučnu karticu Iako de je zvuk sa CD-ROM-a mogao bili prosleđen mikserom i obrađivan i zatim pušten ne isb muzički uređaj kao I kartica

Lepe zamisao ali što se mog mišljeuje biće apsolutno pogrešna. Kulturnije bi bilo da se uvede neki dodatni priključak koji bi prihvatao zvuk sa CD a pa makai to značilo i meli hardverski zahvat u smislu ubacivanja dodatnog kačiča između samog CD uređaja i postojećeg kontrolera Tako sada imamo IDE-SCSI I SCSI 2 adaptere posebno za hard diskove koji se mogu posvađati (a I bespotrebno troše adresni prostor) sa onim ne zvučnoj kartici U svakom slučaju šta je tu je Nekome se može svideti što može proći jeftinije Ili što može ubiti dva muve jedum udarcem eli mi se to Ipak ne sviđa

Borba na tržištu se vodi i po pitanju priloženih ka blova zvučnika mikrofona MIDI box-ova I dodataka koje se mogu dokupiti MIDI box-ovi (opal) Surrouud moduli veza sa telefonom itd Takođe tu je I pitanja priloženog softvera Svako će moći da vidi šte mu tre ba I najvše odgovara

Da pomenem da postoji I VGA (video!) kartica še objedinjenom zvučnom karticom iz ATI-ja Redi se o VGA stereo F/X koje što se zvuka tiče ima osobine ne trećini puta od *Sound Blaster 2.0* do *Sound Blaster Pro*

NEKE OSOBINE NEKIU KARTICA

Na tržištu danes postoji veliki broj zvučuih kartica sa PC-ja. Moram reći da se većina njih prvo hvali kompatibilnošću sa uekim od *Sound Blastera* pa tek onda unapređenjima u odnosu na njega Je nisam imao priliranu vremena da isppolagaunj da se poza blavim pronalaženjem i testiranjem uakih egzotika ta ko da ću se ograničiti samo ua one kod nas najpopu larnije Te kartice se mogu naći kod nas pa sam zato priložio okvirne cene i ime prodavca svake od njih Smatram da tabela pokazuje dovoljno karakteristika da biste se mogli aveutualno odlučiti za neku ili pak atvoriti sebi uvid u stanje na tržištu Među pomenutim inostranim kartama je i jedna naša koje u vreme pisa uje tekstа nije još bila potpuno završeua Iako da sam pukazao samo projektovane karakteristika koje su porad svega jako realne jer se od starta pazilo na sve potrebne delove Redi se o *Pingvin SoftSound* kartici koju je projektovao autor ovog laksia u sarad nji sa Radio Pingvinom i ComTrad-om Ova kartica je dosta specifična i nije predviđena za igre niti za kompenovauje Nameujena je strogo određenom tržištu koje se može uazreti iz pukazanih karakteristika Pomeuuću da su glavni ciljevi i problem tekui razvoja bit 44 1 kHz / 16 bitni / stereo zvuk sinhronizacija i reprodukcija veliki FIFO buffer radi rasterećivanja glavnog procesora i posebno šum (pomanuto „žuborenje" i „grgoljenje" pri raznim aktivnostima računara) koji se prenosi preko uglavnom preko mase

U novije vreme kao što je to ranije bilo i sa VGA i SVGA karticama, postavije sa pitanja da li kartica koristi samo XT-ISA bus ili ma i mogućnosti 16-bitnog pristupa preko za to još uvek dovoljno brza AT-ISA magistrala Novija kartice podržavaju 18-bitni pristup i 16-bitna DMA kanala, ali moram napomenuti da dosta samih matičnih ploča u našim računarima ne trpi 18-bitni DMA transfer iz meni za sada nepoznatog razloga.

Još jedno objašnjenje u vezi se tabelom Uz *Sound Blaster 16* seriju se ovoil i *daughterboard* koji predstavlje izvršan MIDI player da ge us nazovem box jer je u pitaju kartica koja se priključuje NA neku *Sound Blaster 16* karticu i pri tome joj daje izvanredno realuu simulaciju nekih 256 instrumenata (192 prava + 63 ugrađene kombinacije) uz mogućnost dow-uloadi a još 192 instrumeuta Ovaj izvršan dodatak, *Wave Blaster* drži samplove instrumeuata u 4 Mb (da dva čipa od po dve megabajta) EPROM e s sva ra uglavnom uz pomoć jednog velikog E-mu čipa I male Motorole 68000 ne 16 MHz preko profesionalnih MOSSEL 16-bitnih D/A konvertora Stereo naravno Treba reći da ovo nije skrivalenl MIDI klu kod starih Blastera već nadgradnja samog SB16 koja vam i de-lje dozvoljava da kupite eksterni MIDI kil I na ujega recimo uakačka MIDI Blastar Redi se o mlemom MI DI procesoru sa apsolutno realuim zvukem izvanredno kvalkata.

NIJE SVE U HARDVERU

Hardver nije sve što je bitno za dobar zvuk Recimo kade uporedimo bilo koju Amigu (pa i 500) sa PC-jem I PC karticama, možemo reći da Imaju bedan do jednako kvalitetau zvuk Međutim ne Amigi kao mašini za igre razvijeni su razni programi koji izvlače maksimum njeuih mogućnosti Došlo se i do standarda za muzičke module koji su kesnije prešli I na PC-ja Uz takav softver PC dodija još bolji zvuk

Dobar primer su I razni pušlači semplova i modu le Ime ih koji I bez dodatne zvučne kartice mogu da izvuku zavidan kvalitet iz PC-ja na način koji sam ra nije opisao Na ovaj način se može izvući nekih 5-6 bita na bilo kom PC-ju (de i na XT-u – probajte IM-PULSE koji sam na XT-u i pravio) dok se uz najnovi joj generaciji PC-ja može može postići i više – time ću sa baviti u skoroj budućnosti ako budem imao vre mene

Najdrastičnije razlike se ogledaju u MIDI sekven cer program ma i editorima semplova. Sređom tolike razlika olakšavaju izbor pa se lako dolazi do progra me koji je odgovara svrsi

Softveri koji koustl zvuk uglavnom se svodi na dve grupe Prva su razni sekvenceraki (MIDI) programi ko je koriste uglavnom samo ljudi od muzike Ovaj sof tver zahteva MIDI kartice dok neki mogu da rade i sa FM sintetizatorom. Druga su igre One opet uglavom koriste samo FM sintetizator poneka se ohiabri i da pusti ušeto digitalnog zvuka dok ima i onih koje mogu da sviraju i preko MIDI interfejsa Možemo reći da su tu I razni puštači modula ali tu je muzika sama sebi svrhe Muzički moduli koriste iskljućivo D/A konvertor ski deo na zvučnim karticama i uz ujega što se muzi ke tiče. Izviače skoro maksimum Oni u neku ruku sof tverski simuliraju ono što bi se odigralo u nekom MIDI player-u a tom razlikom što se ovde dopuštle maksi malno 4 da 8 kanala. Za igre ja i to dovoljno Retko se dešava da ljudi zainteresovani za programiranje i hardver uđu I u svet muzike i pozabave se ozbiljno razvojem ili prepoznavanjem istog Za njih gotov sof tver uglavnom ne zvuči uišta Ili znači vrlo malo Uglavnom ge prave sami Oni (tu sam I ja) predstav ljaju još jedan krug konsnika, ali sa relativno „egzotič nim" potrebama pa ih neću dalje komentarisati

Kada (i ako) se budete odlučivali za neku od zvučnih kartica obratite pažnju na to koje programe dobijate uz njih Uz neke dobijete potpuno nekvalite tan softver dok uz neke druge dolaze programi od mah upotrebljivi

ŠTA JE ZA KOGA?

Ako želite saval od mene mogu vam iskreno preporučiti tri kartice zavisno od toga kojoj grupi korisnika pripadate Ako ste slovremeno u više grupa, uzmite onu bolju (i skuplju) karticu Za igre je dovoljan običan *Sound Blaster* ueki ujegov klon Ili *Sound Galaxy NX* Ono što ćete čuti iz njih je uglavnom FM muzika i ponekl put neki kratak sampl Mali je broj igara koji koristi mogućnosti karbca nove generacije Ako želite malo više tu je *Sound Blaster PRO* Što se tiče *Sound Blastera I Windows*-a, razmišljajte o verzijama od *2.0* na dalje U principu ako možete malo da sačeka kupite još para, uzmita neku 16 bitnu karticu jer u ovom slučaju sredine može vrlo brzo da se poka že samo kao malo pozlaćena. Za profesionalce nema dilema kombinacija *Sound Blaster 16ASP* sa *Wave Blasterom* Uz ujih se dobija toliko softvera da možete odmah da krenete punom parom *Wave Blaster* će vam svirati sve što poželite a ako tome dodate koju klavijaturu i par uređaja se strane Inače *Sound Blaster 16* sa još uvek relativno teško nabavije kod nas, a za sada ga jedino ima u ComTrad-u *Pro Audio Spectrum* jeste dobra kartica, ali je ne preporučujem zbog pojave *Sound Blaster 16* serije sa manjom ce nom za sva varijanta a isim ili boljim karakteristika ma u svim pogledima – izgleda da se Creative Tec hnologies naoštrio I krenuo drugačijom polrikom.

Što se tiče upotrebe u razim studijima, radio-difuzn im organizacijama i sličnim okolnosbma, gde je po trebno rediti sa ogromnim semplovima I puštanjem di rektno sa hard diska mogu same da preporučim Pingvin SoftSound sistam koji ja za to tržište namenjen i cenom i mogućnostima.

Kod PC ja postoji i taj problem da se većina progra ma pravi za čvisto tlo odnosno ze već dobro uhodaue i standardne kartice Tako sada imamo gomilu programa koji podržavaju običan *AdLib* običan *Sound Blaster* Retko šta (osim originalnog softvera, narav no) podržava bilo šta iznad onoga što nudi običan *Sound Blaster* Jedino *Windows* zbog uvođenja mu zičkih drajvera pretenduje da u potpunosti iskoristi mogućnosti svih kartica. Pitanje je samo koliko vi ko uspiše sam *Windows* i koliko u ujemu koristite zvuk. Dok za obične konsnika i programerske entuzijaste postoji više interesaunih kartica koliko zbog cene to liko I zbog mogucnosti tolko za profesionalce nema dilema njima uvek i taba najbolje – je preporučujem *Sound Blaster16ASP - Wave Blaster*. U svakom slu čaju koliko para toliko muzika

VIŠEPROGRAMSKI RAD

Nastavljajući proučavanje mikroprocesora 80486, ovoga meseca se bavimo višeprogramskim radom, smenom procesa i počinjamo sa opisom njegovog složenog interapt mehanizma.

Dejan Ristanović

„Klasičan" višeprogramski rad koji 80486 podrža va zasniva se na smeni procesa – procesori u jednom trenutku može da izvršava samo jedan program ali se čestom zamenom izvršavanih programa stvara utisak da više njih radi istovremeno. Svaki od programa u sistemu zapravo „ima utisak" da je čitavo vreme procesori bio dodeljen samo njemu i nema informaciju o tome da je više puta prekidan i reaktiviran

PROCESI I POSLOVI

Umesto fraze „program koji se izvršava" ispravni je bi bilo koristiti reč „proces" ili „posao" jer se smena programa ne svodi samo na zamenu koda – svi podaci kojima je program raspolagao takođe se moraju odložiti i zameniti podacima drugog programa, a čuvaju se i razne kontrolne i statusne informacija koje omogućavaju docniji nastavak rada, na primer sadržaj registara. U terminologiji operativnih sistema kaže se da se pri zameni poslove čuva „kontekst" tekućeg programa što obuhvata sve pomenute parametre. *Intel*-ova dokumentacija ne insistira na terminima „proces" „posao" i „kontekst" već pominje samo „program" ali samo mi povremeno koristit ove izraze kako bi objašnjenja bila preciznija

Osnovna struktura na osnovu koje 80486 ostvaruje višeprogramski rad je TSS *Task State Segment*. Kao i ostali objekti u sistemu TSS ima svoj deskriptor i precizno definisanu strukturu sa slike 1. Primetićete da je TSS prilično sličan memorijskom segmentu koji smo opisivali pre pet meseci s tim što je njegov S bit postavljen na nulu što kao što smo lada videli označava TSS. Polje TYPE sadrži vrednost 1001 ili 1011 (binarno). Nekadašnji *variable bit* sada se zove *busy bit* i postavlja se na jedinicu kod procesa koji se trenutno izvršava kao i kod onih procesa koji su inicirali izvršavanje tekućeg procesa – time se ostvaruje neka vrsta lanca tekućih poslova. Pokušaj ponovnog pokretanja procesa koji je već markiran kao tekući (*busy*) izaziva odgovarajući interapt „Sive zone" slika 1 predstavljaju podatke koje su konstruktori 80486 smatrali za retko promenljive pa su u cilju uštede vremena, izbegli njihovo beleženje u TSS prilikom zamene procesa. Ipak nove vrednosti tih polja se upisuju iz TSS a novog procesa što znači da operativni sistem mora voditi računa o tome da li se neka od ovih vrednosti promenila i ako jeste naknadno je ažurirati u TSS-u

Neka od značajnih polja TSS-a su SSn ESPn (sliek pointen prestena *n* – više o njima pročitali ste u „Računarima 93"). CR3 (fizička adresa memorijske stranice koja je dodeljena procesu) LDTR (selektor LDT a (*local descriptor table*) procesa) T (*trap bit* ako je setovan svaka zamena procesa izaziva *debug* interapt) i I/OP (šesnaestobitni oblest koji određuje poziciju mape dozvoljenih ulaza-izlaza (više o njoj docnije) u okviru TSS-a vrednost 0 označava de mapa ne postoji). Deo TSS-a predviđen je za informacije specifične za dati sistem po rasporedu koji definiše tvorac operativnog sistema a iza svega je eventualna mapa dozvoljenih ulaza izlaza

Tekući proces je određen sadržajem *task* registra (TR) – operativni sistem pri podizanju definiše ovu vrednost i dode se ona tokom čitavog rada računara menje pod kontrolom podsistema za zamenu procesa. Neki početnici „naivno" misle da se promenom sadržaja TR-a zamenjuje aktivni proces – videćemo da je to tek delić komplikovanog mehanizma zamene koji se obavlja kada je sve drugo već pripremljeno. Nekontrolisana promena sadržaja TR verovatno će dovesti računar u nedefinisano stanje koje vodi u pot puni prekid rada žalo pravo na tu promenu ime sa mo jezgro operativnog sistema

„Puna" zamena procesa obuhvata smeštanje svih relevantnih parametara u TSS tekućeš procesa, punjenje registara vrednostima iz TSS-a procesa kome treba predati kontrolu i upisa njegovog selektora u *task* registar. Deskripton svih TSS-ova mogu se locirati preko tabele GDT (*global descriptor table*) čiju smo strukturu upoznali u prvim nastavcima ove serije tekstova

SCENARIO ZAMENE

Zamenu procesa inicira jedan od četiri važna događaja izvršenja instrukcije skoka (daleki CALL ili JMP) na lokaciju čiji selektor ukazuje na TSS deskriptor skok na lokaciju čiji selektor ukazuje na *task* kapiju izvršenje IRET-a koji zahteva prelazak na prethodni task (NT flag setovan) i najzad bilo koji interapt preko vektora koji ukazuje na *task* kapiju. Ukratko rečeno mikroprocesor će izvršiti zamenu posla samo kada je to neophodno tj. kada se na neki način „skače" u zonu koja ne pripada tekućem poslu. Da bi se obezbedila regularna zamena poslova koji se „mimo" izvršavaju hardveri povremeno generiše interapt pa zamena procesa nastaje po poslednjoj pomenutoj osnovi

Kada do nje dođe zamena procesa odigrače se u najviše 10 faza

1. Ukoliko do zamene posla na dolazi usled hardverskog ili softverskog interapta odnosno IRET instrukcije proveravaju se privilegije DPL polje deskriptora procesa ili kapije mora biti numerički manje od CPL-a tekućeg posla i RPL polja selektora. Za detaljan opis mehanizma zaštite pogledajte „Računara 93"
2. Proverave se da li je čitav TSS odlazećeg procesa u memoriji i može li se u njega upisati 104 bajta statusnih informacija. Ako li uslovi nisu ispunjeni nastupa interapt i kontrola se predaje jezgru operativnog sistema koje treba da obezbedi potrebnu memoriju izbacujući neki dugo nekorišćen segment na disk
3. Proverava se da li je čitav TSS procesa kome treba predati kontrolu u memoriji. Ako nije na stupa interapt i jezgro operativnog sistema dovlači te podatke sa diska. Ako je sve u redu sadržaji svih registara se popunjavaju odgovarajućim poljima TSS-a novog procesa ako se vrednost CR3 promenila TLB (*Translation Lookaside Buffer*) se prazni. Time su svi registri opšte namene i segmentni registri popunjeni ispravnim vrednostima, ali *shadow* registri i dalje nisu definisani. CS na primer može da ima vrednost 41AFh ali deskriptor koji odgovara selektoru 41AFh ne mora biti u memoriji što znači da bi prost nastavak rada mogao da izazove kioh
4. Uspostavljaju se sve potrebne veze između novog i starog posla. Ako do zamene poslova dolazi zbog JMP instrukcije TSS deskriptori odlazećeg posla markira se kao „slobodan" (*busy* bit se postavlja na nulu) dok se *busy* bit dolazećeg posla postavlja na jedinicu. Ako do zamene posl'ova dolazi usled interapta ili CALL-a odlazeći posao i dalje ostaje označen kao *busy* a kao takav se markira i novi posao uz to se postavlja NT–1 (registar EFLAGS) i ukazatelj na prethodni posao upisuje u *back link* polje novog TSS-a. Ukoliko je najzad do zamene došlo zbog instrukcije IRET odlazeći posao se markira kao slobodan (*busy*=0)
5. TS bit registra CR0 se setuje i podešava se polja privilegije tekućeg posla na osnovu RPL polje odgovarajućeg selektora.
6. Ukoliko LDTR sadrži ispravan selektor dalnišu se LDTR *shadow* registri. Ukoliko je LDTR=0 ovaj korak se preskače a ako selektor postoji ali je neispravan nastupa interapt
7. Deskripton CS SS DS ES FS i GS se (tim redosledom) upisuju u *shadow* registre. Svaki od deskriptora se testira kako bi se proverila prava pristupa i markira se kao prisutan u memoriji. Ako nešto od toga ne uspe nastupa interapt
8. *Local enable* bitovi DR7 se postavljaju na nulu
9. Ako je T bit TSS-a odlazećeg procesa setovan, nastupa *debug fault*.
10. Počinje izvršavanje novog posla od instrukcije na koju ukazuje CS:EIP. Novi posao se izvršava sve dok ne bude prekinut iz nekog od četiri navedena razloga. I ta smena se ponavlja sve do isključenja računara

MAPA DOZVOLJENIH ULAZA-IZLAZA

Ulazno-izlazne informacije su privilegija samo nekih od procesa koje 80486 izvršava – da bi neka ta kve operacija bile dopuštena IOPL polje registra EFLAGS mora imati odgovarajuću vrednost ili ta operacije mora biti naknadno odobrena na osnovu mape dozvoljenih ulaza izlaza. Preciznije rečeno IOPL određuje najmanje privilegovan posao kome se ipak dopušta da izvršava ulazno-izlazne operacije bez ikakvih restrikcija ako je na primer. IOPL=2 samo programi koji se izvršavaju u nultom prvom i drugom prstenu smeju da izvršavaju IN i OUT. Ako bi neki program iz trećeg prstena pokušao da izvrši ulazno-izlaznu operaciju, bila bi neophodna dodatna provera bazirana na mapi

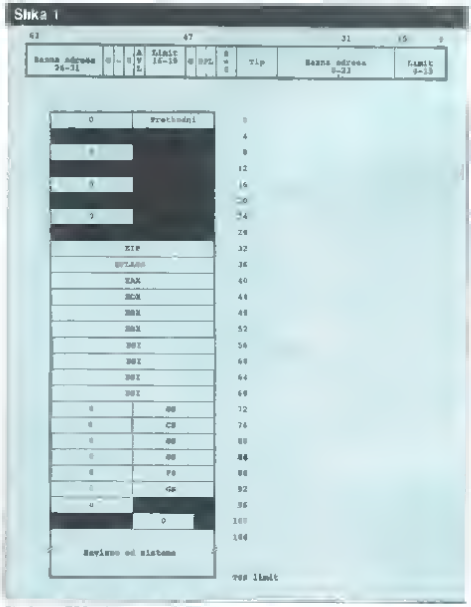
Maksimalna dužina mape dozvoljenih ulaza-izlaza je 8192 bajta pri čemu svaki bit odgovara jednom od 65536 postojećih portova. Ako je neki bit setovan task ne može da pristupa odgovarajućem portu (pristup izaziva *general protection fault*) i obrnuto. Na taj način se nekim niže privilegovanim procesima mogu odobriti određene ulazno-izlazne operacije (na primer pristup tastaturi i ekranu) dok su im druge, na primer direktan pristup disku, trajno zabranjene. Moderni operativni sistemi ovo pored ostalog, koriste i kao veoma uspešan mehanizam borbe protiv virusa – me koliko autori nekog virusa bio vešt prodor u sistem mu neće biti moguć ukoliko je vlasnik dovoljno oprezan da nove programe izvršava se nekog nepriviligovanog konsničkog imena. napad će zaustaviti kontrolni mehanizmi samog mikroprocesore

Mapa dozvoljenih ulaza-izlaza na mora da bude dugačka osam kilobajta – njen kraj je određen *limit* poljem TSS-a. Svi portovi koji nisu definisani u mapi smatraju se nepristupačnim za nepriviligovane procesa

INTERAPT MEHANIZAM

U dosadašnjem tekstu smo često pominjali reč „interapt", podrazumevajući pod njim prekid izvršava-

Slika 1



Struktura TSS-a i njegov deskriptor

Slika 2

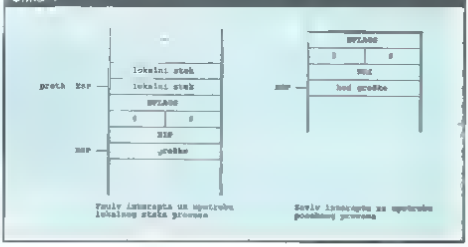
Broj	Tip	Opis
0	Fault	Deljenje nulom
1	Fault/Trap	Debager
2	Interrupt	Nemaskirani interrupt (NMI)
3	Trap	Prekidna tačka (breakpoint)
4	Trap	Prekoračenje opsega (overflow)
5	Fault	Prekoračenje granica niza
6	Fault	Neispravna naredba
7	Fault	Koprocesor nije raspoloživ
8	Abort	Greška tokom obrade greške (double fault)
9	Abort	Prekoračenje koprocesorskog segmenta (samo 80386)
10	Fault	Neispravan TSS
11	Fault	Segment nije u memoriji
12	Fault	Prekoračenje steka
13	Fault	Prekoračenje pravo pristupa
14	Fault	Stranica nije u memoriji
15		Nerezervisano za Intelove mikroprocesore
16	Fault	Greška koprocesora
17	Fault	Neuravnata struktura alignment check (samo 80486)
18–31		Nerezervisano za buduće Intelove mikroprocesore
32–255	Interrupt/Trap	Softverski prekidi

Svi softverski prekidi (exceptions) mikroprocesora 80386/80486

Slika 3

Deskriptor kapije koja obrađuje hardverski prekid i trap

Slika 4



Interapt stek uz i bez aktiviranje novog procesa

nje programa i privremeni prelazak na neku drugu proceduru posle koje će biti nastavljen prekinuti program tačno od onog mesta na kome je interapt nastupio. Reč interapt se međutim često koristi u užem smislu i označava samo prekid koji je iniciran hardverskim signalom preko INTR (maskirani) ili NMI (nemaskirani interapt). I sam *Intel* koristi reč interapt isključivo u tom užem smislu. Hardverski interapti nastupa ako je NMI nožica na logički niskom nivou dok je za maskirane prekide „nadležne" kako INTR nožica tako i IF bit EFLAGS registra. Ako je IF=0 interapti su maskirani i 80486 ih ne „primećuje". Ukoliko je IF=1, mikroprocesor će generisati takozvani *interrupt acknowledge* signal a periferija mu odgovara brojem interapta koji je nastupio broj treba da se nalazi u intervalu [32 255].

Svi oni ti prekidi se u *Intel*-ovoj terminologiji nazivaju *exception* – ne znamo kako bi se to pogodno prevelo pa ćemo koristiti izraze „hardverski prekid" (*interrupt*) i „softverski prekid" (*exception*) videćemo da je tretman hardverskih i softverskih prekida praktično identičan što možda opravdava korišćenje istog izraza. Softverski prekidi su takođe podeljeni na grupe *traps faults* i *aborts* – bitna razlika između ovih grupa je u trenutku njihovog detektovanja. *trap* se detektuje **nekon** izvršavanje instrukcije *fault* **u toku** izvršavanja, dok je *abort* nedefinisano stanje koje može nastupiti bilo kada i posle koga obično nema nastavka tekućeg programa.

U *trap*-ove spadaju uglavnom instrukcije tipa INT kojima korisnik eksplicitno zahteva izvršavanje određene procedure obično da bi zatražio neku uslugu operativnog sistema kao što je ispis teksta na ekran ili komunikacija sa diskom. Za razliku od prvih *Intel*-ovih mikroprocesora 80486 je veoma selektivan kada treba „odobriti" izvršavanje neke INT instrukcije ukoliko na primer program koji se izvršava u trećem prstenu izvrši INT 45 deskriptor IDT(45) more da ima DPL=3 inače će nastupiti *protection fault*. Na ovaj način se nepnvilegovanim programima onemogućava korišćenje bilo kog interapta kako bi se sprečio nekontrolisani skok u neku od rutina koje obrađuju hardverske prekide i koji se obično nalaze u multom prstenu jezgru operativnog sistema. Obzirom da je tekuća instrukcija pre *trap*-a već izvršena, na stek se stavlja adresa **sledeće** instrukcije od koje će se izvršavanje programe nastaviti.

Greška (*fault*) nastupa ako procesor **tokom** izvršavanja instrukcije primeti da je ne može uspešno okončati. Često se na primer dešava da operand instrukcije bude u nekom segmentu koji je privremeno prenesen na disk (koncept virtuelne memonje) ili da izvršavanje instrukcije prevazide prava tekućeg programa. Na stek se tada stavlja adresa tekuće instrukcije i prelazi se na odgovarajući potprogram operativnog sistema koji treba da konguje grešku i instrukcijom IRET naloži nastavak programa tj. **ponovljenje** tekuće instrukcije ili, eventualno da suspendu je program.

Fatalne greške (*Abort*) označava da je deo konteksta tekućeg procesa nepovratno oštećen iz razloga koje je obično nemoguće odredit (možda neispravan hardver?) Očekuje se da operativni sistem preki ne tekući proces (nije predviđen njegov nastavak) i informiše korisnika o događaju koji je nastupio. Moguće su i „katastrolalne greške" tj situacije u kojima procesor naprosto nije u stanju da nastavi izvršavanje nekog softverskog prekida takozvana „greška u greški" (*double fault*). Na jednostavnijim operativnim sistemima dvostruka greška izaziva prekid iade procesore i prema tome totalni pad sistema. Ozbiljniji operativni sistem će pieusmeti *double fault* prekid (interapt 8) i prekid usled neispravnog TSS-a (Intelapt 10) ne odvojene poslove (IDT 8 i IDT 10 treba da budu *task* kapije) što će naterati procesor da za svaki softverski prelod učita novi kontekst za slučaj da je kontekst tekućeg posla oštećen na način koji bi „zbunio" i samu rutinu za obradu greške. Ukoliko ni tada oporavak ne uspe nema druge nego obonti čitav računar – verovatno se radi o hardverskom problemu ili teškom bagu samog operativnog sistema.

Slika 2 sumira sve softverske prekide mikroprocesora 80486

INTERAPTI I KAPIJE

Pozivi potprograma koji opsluzuju pojedine prekide obavljaju se prako kapija što je i normalno ako se zna da su standardno u pitanju segmenti operativnog sistema koji su privilegovanij od konsnčkog programa. Deskripton svih interapt kapije organizovani su u IDT *Interrupt Descriptor Table* – nema nikakve razlike u struktun između globalne tabele deskriptora (GDT) i IDT-a.

Kada nastupi bilo koji prekid kontekst tekućeg procesa se upisuje u postojeća TSS a zatim se u registre upisuju podaci iz TSS-a koji je povezan sa interapt kapijom. To znači da rutina koja obrađuje prekid može da ima svoj adresni prostor pa čak i sopstvenu lokalnu tabelu deskriptora (LDT). Time je rešen problem „nekontrolisanog" rasta steka osnovne aplikacije zbog učestalih interaptove – sada autor programa može da proračuna koliko mu je **stvarno** prostora na steku potrebno i naravno uz ostavljanje neke male bezbednosne margine prestane da brine o nepredviđenim događajima u sistemu nepredviđeni interapt može samom sebi da obezbedi potreban stek! Pošto svako unapređenje ima svoju cenu, autor operativ-

Nastavlja se na strani 60

ŠESTICA IZNUTRA

Najinteresaantnija osobine nedavno predstavljanog MS-DOS-a 6 su optimizacija memorije, *MemMaker*, i tehnologija za transparentnu kompresiju diska, *DoubleSpace*. Programeri koji su razvili MS-DOS 6 – Bendžamin V. Slivka, Erik Strob i Ričard Fridmen (Benjamin W. Slivka, Eric Straub i Richard Freedman) – govore o softverskim tehnikama koja su koristili u najnovijoj verziji već legendarnog Microsoft operativnog sistema.

MemMaker automatski skenira sve drajvere uređaja i TSR programe u datotekama AUTOEXEC.BAT i CONFIG.SYS, skenira prostor gornje memorije i određuje najefikasniji način za učitavanje ovih datoteka u blokova gornje memorije. Novi EMM386 omogućava da *MemMaker* oslobodi do 100 KB dodatnog prostora gornje memorije.

DoubleSpace pomoću kompresije podataka, efikasno udvostručuja raspoloživi prostor na disku. On radi u pozadini, vršeći dekompresiju datoteka onda kada aplikacije to zahtevaju a zatim ponovo kompnmuje datoteke kada se smeštaju na disk.

Obe tehnologije predstavljaju korak naprad za MS-DOS-a jer integrišu upravljanje memorijom i kompresiju datoteka u samo jezgro operativnog sistema.

MemMaker

MemMaker smo projektovali sa namerom da postignemo tri cilje: sigurnost (ili pouzdanost) lakoću korišćenja i optimalnu upotrebu memorijskih resursa. Kada je trabalo praviti kompromis dali smo prioritet sigurnosti i lakoći korišćenja a žrtvovali optimizaciju.

Tokom optimizacije memorije *MemMaker* dva-put iznova podiže sistem. U prvom prolazu traga da ustanovi obim svih drajvera uređaja i programa u datotekama CONFIG.SYS i AUTOEXEC.BAT (to obavlja tako što svaku liniju u kojoj se poziva drajver uređaja ili program dodaje komandu SIZER). U drugom prolazu se ovi drajveri i programi učitavaju u optimalnu konfiguraciju gornje memorije. Naš primarni cilj je bio sigurnost ali mislimo da je nemoguće projektovati optimizator memorije koji nikada neće zakazati pošto ime programe i drajvera uređaja koji ne funkcionišu ispravno kada se učitaju u gornju memoriju. Rešenje je bila primena "modela stanja" (*state model*) koji deleži proces optimizacije i omogućava da se *MemMaker* elegantno oporavi ako se sistem zaglavi za vreme procesa optimizacije.

Ukoliko *MemMaker* greškom mapira kritičan deo memorije koji već koristi ROM na nekom adapteru sistem može pasti pri ponovnom startovanju. Model stanja *MemMakera* automatski otkriva ovu situaciju i dopušta vam da poništite sve promene ili pokušate se konzervativnijim algoritmom za mapiranje koji stvara manje gornje memorija ali uspešno izbegava pomenuti konflikt. *MemMaker* se oporavlja na sličan način ukoliko drajver uređaja ili program zaglavi kada se učita u gornju memoriju (ili ako se to dogodi zbog pokušaja nekog drugog drajvera ili programa da se učita u gornju memoriju).

Pri svakom izvršavanju *MemMaker* kreira MEMMAKER.STS datoteku u kojoj čuva informacije koje prikupi tokom optimizacije – na primer podatke o veličini drajvera. MEMMAKER.STS sedrži liniju STATE= u kojoj se beleži trenutno stanje procesa optimizacije *MemMaker* modifikuje ovu liniju više puta tokom optimizacije: na primer, kada završi prvi prolaz.

Pri svakog podizanja sistema *MemMaker* ubacuje red DEVICE=C:\DOS\CHKSTATE.SYS /S STATEARG u prvu liniju datoteka CONFIG.SYS. STATEARG definiše stanja koje se javlja u MEMMAKER.STS kada se učita CHKSTATE.SYS i kada je sve u redu. Ukoliko CHKSTATE.SYS otkrije neslaganje između STATEARG i stanja u MEMMAKER.STS pri učitavanju *MemMaker* zaustavlja optimizaciju i pitalazi u akciju. Pošto je CHKSTATE.SYS prvi drajver koji se učitava u CONFIG.SYS on se može učitati i preduzeti potrebne korake i kada se ponovno podiže sistem nakon pada. Posle završetka optimizacije *MemMaker* uklanja liniju CHKSTATE.SYS iz datoteka CONFIG.SYS.

IDEALNA RAVNOTEŽA

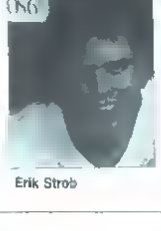
Proces optimizacije koji obavlja *MemMaker* ne konfiguriše automatski 32 KB prostora gornje memorije od B000-B7FF i to iz dva razloga: ovaj prostor koriste Super VGA ploče a *MemMaker* ne može uvek da otkrije prisustvo SVGA kartice.

Nismo ugradili ni *Flex Frame/Stealth* opciju koju nuda neki drugi programi za upravljanje memorijom. One omogućava menadžeru memorije 386 da učita drajver uređaja ili TSR u blok gornje memorije (UMB) koji inače na bi bio dovoljno prostran da ih primi. Ove opcije koristi umjenicu da je inicijalni oblik većine drajvera i TSR programa veći od konačnog.

Razlika između početna i konačne veličine obično nastaje zbog koda za inicijalizaciju koji se kasnije može odbaciti. Trik je u tome da se detektuje memorijski prostor u ROM-u ili RAM-u nekog od adaptera koji graniči se ciljnim blokom gornje memorije a nije zauzet dok se taj drajver učitava. Ako menadžer memorije nađe takav susedni prostor on ga privremeno pozajmljuje učitava drajvar uređaje i inicijalizuje ga. Kada se drajver ili TSR sažmu toliko da mogu stati u taj blok menadžer memorije vraća pozajmljeni prostor njihovim vlasnicima, ROM u ili RAM u na adapteru.

Nevolje kod ovakvog postupka je u tome što se može desiti da kasnije dodate neki drajver TSR ili karticu koja će onesposobiti odabrani susedni deo memorija. Ukoliko lada ponovo ne izvršite optimizaciju sistem će verovatno zaglaviti a biće teško poslavrti dijagnozu.

Ričard Fridmen "Sigurnost i lakoća korišćenja su bili prioritetni u dizajnu *MemMaker*-a"



Erik Strob

Bendžamin V. Slivka

DoubleSpace

DoubleSpace koristi transparentni Lampel-Ziv kompresioni algoritam koji udvostručuje – ili gotovo udvostručuje – raspoloživi prostor na tipičnom hard-drajvu. *DoubleSpace* to obavlja vodeći maksimalno računa o kompatibilnosti, lakoći korišćenja i sigurnosti podataka. Pošto je kopresija diska tehnologija zasnovana na sistemu datoteka, cilj nam je bio da integrišemo *DoubleSpace* u sistem datoteka MS-DOS-a.

U srcu *DoubleSpace*-a je LZ kompresija. Lampel i Ziv su autori čuvenog članka LZ77 o kompresiji bez gubitka podataka (vidi "A Universal Algorithm for Sequential Data Compression" u časopisu *IEEE Transactions on Information Theory* vol IT 23 broj 3 maj 1977) u kojem dokazuju da se može postići gotovo optimalna kompresije bez poznavanja tipa podataka koji se obrađuju. Sasvim jednostavno ukoliko se nađe na ponavljanje, ono se obeležava kao referenca na prethodno pojavljivanje u bloku istih podataka.

Kao primer pogledajmo sledeći string:

```
1234567890123456789012345678901234567890123
The rain in Spain falls mainly on the plain
```

The rain in Spain falls mainly on the plain

Prama LZ77 ovo se može predstaviti u kompnmovanom obliku tako što ćete ispisati kombinaciju eksplicitnih bajtova i podudarnih delove (*matches*) koji se pokiapaju sa prethodnim podacima. Za navedeni primer dobili bismo:

The rain <3,3>Sp<9,4>falls m<11,3>ly on t<34,3>pl<15,3>

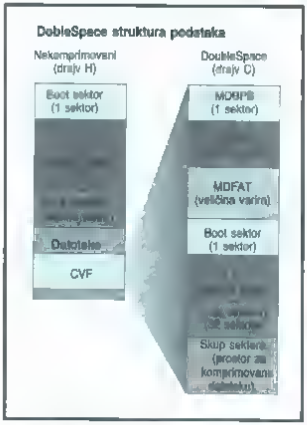
Ovde <*ofset,duzina*> pokazuje da se treba pomeriti unatrag za *ofset* bajtova u stringu i sa log mesta kopirati onoliko bajtova koliko piše u parametru *dužina*. Ili drugi primer: ako kompnmovani podaci izgledaju poput 0<1,200>1<1,100>2<1,50>, dekomprimovani će imati oblik 0 01 12 2 odnosno dobićemo 201 0 (nulu) 101 1 (jedinicu) 51 2 (dvojku).

Da bi se kreirala LZ77 shema pogodna za *DoubleSpace*, konstruisali smo algoritam koji brzo pronalazi ponavljanja i zauzime vrlo malo prostora. Pošto je *DoubleSpace* deo svakog čitanje i zapisivanja ne kompnmovanom drajvu on mora biti brz. A pošto je stalno razidenjan mora biti mali. Trebalo je da odeberemo shemu za kodiranje koje daje dobru kompresiju ali se lako kodira i dekodira.

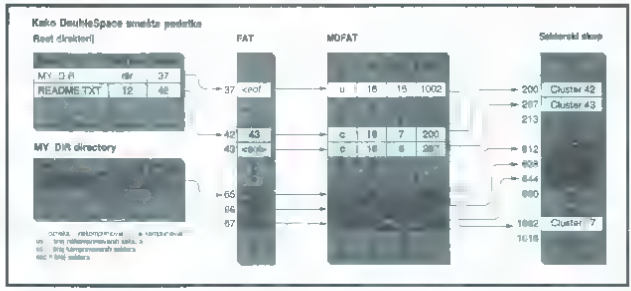
Konačni rezultat je dobar kompromis. *DoubleSpace* u proseku smanjuje performanse za 5 do 15% u odnosu na nekompnmovani drajv. Ali retku su korsnici koji žo zaista primećuju. *DoubleSpace* zauzima 43 KB memorije i može se učitati u visoku memoriju. Slepan kompresnje se kreće od 1.4 do 1 za izvršne datoteke 2 ili 1 za radne tabele i tekstualne datoteke pa sve do 3-1 za bit-mape i druge redundantne datoteka.

CVF PRISTUP

Kao i mnogi komercijalni programi za transpareninu kompresiju podataka *DoubleSpace* smešta



DoubleSpace koristi latu strukturu podataka kao i nekomprimovani hard drajv ali dodaje nekoliko novih struktura, koje omogućavaju da se podaci



Na *DoubleSpace* drajvu, datoteke i poddirektoriji se mapiraju kroz FAT i MDFAT do skupa sektora. *DoubleSpace* smešta datoteke po sektorima a ne po klasterima, a MDFAT služi kao veza između nekomprimovanih logičkih klastera u FAT-u i komprimovanih podataka u fizičkim sektorima sektorskog skupa. README.TXT zauzima dva klastera koja se se 32 sektora sažimaju na 13 u sektorskom skupu. BRICK.ZIP sa ne može komprimovati, on se u MDFAT-u registruje kao nekomprimovan i zauzima punih 16 sektora po klasteru

sve komprimovane podatke u jednu datoteku – koju nazivamo CVF (*compressed volume file*) – ona u stvari predstavlja poseban logički drajv. Običan FAT drajv se može pretvoriti u *DoubleSpace* drajv a proces konverzije se obavlja na licu mesta bez ponovnog formatiranja i odvajanja particija. Sistem može imati do 255 CVF datoteka a jedan CVF može sadržati do 512 MB podataka.

CVF pristup ima nekoliko prednosti nad automatskom kompresijom jedne po jedne datoteke. Pošto smešta komprimovane podatke na nivou klastera MS-DOS ne troši da vrši dekompresiju cele datoteke kada joj direktno pristupa radi čitanja ili zapisivanja. Osim toga, sistem prijavljuje dvostruko veći kapacitet tvrdog diska kada javlja o veličini pojedinačnih nekomprimovanih datoteka. Za razliku od ovoga, kod pristupa "datoteka po datoteka" prijavljuje se isti kapacitet diska kao i ranije.

Pošto se datoteke za razliku od tvrdih diskova prenose sa sistema na sistem različito prijavljivanje veličine kada se dekompriruju i kopiraju na drugo mesto može izazvati zamršene probleme u vezi sa kompatibilnošću. Pošto *DoubleSpace* afikasno simulira strukturu standardnog FAT drajva postojeće disk alatke rade se *DoubleSpace* drajvom. Konačno, pošto *DoubleSpace* skladišti klastera na bazi sektora CVF pristup rešava problem praznih klastara, male datoteke, čak i ako se ne mogu komprimovati sa smeštaju u što je moguće manji broj 512 bajtnih sektora, a ne zauzimaju celi klaster kao na normalnom FAT drajvu.

Ima korisnika koji se pribojavaju da CVF nije tako pouzdan kao normalni FAT laji-sistem, pošto sve "trpa" u jednu daloteku. Međutim CVF nije nimalo podložniji oštećenju od običnog FAT drajva. Oba sadrže FAT rut direktorij, poddirektorije i sektore u koje se smeštaju podaci iz datoteka. CVF ima i dodatnu strukturu podataka (MDFAT) koja obezbeđuje optimalno dodeljivanja prostora.

Kao i kod FAT drajva, jedan neispravan sektor u CVF znači da gubite samo datoteku (ili deo datoteka) koja sadrži neispravni sektor. Oštećen sektor FAT rut direktorija ili poddirektorija je već leži problem, ali lakvim rizicima su podjednako izloženi i FAT drajv i *DoubleSpace* drajv.

DBLSPACE.BIN kernel datoteka se izvršava sa *boot* drajva odmah nakon IO.SYS i MSDOS.SYS. Zatim drajv C postaje CVF, a *boot* voluman postaja drajv H, koji sadrži skrivene *DoubleSpace* datoteke. IO.SYS i MSDOS.SYS su kopirani i na CVF drajvu, radi kompatibilnosti sa aplikacionim programom

UNUTAR CVF

DoubleSpace CVF za imanovanje koristi konvenciju DBLSPACE.*nnn* gde je *nnn* neki broj između 000 i 254 i predstavlja broj CVF sakvence. *DoubleSpace* koristi broj sakvence 000 kada CVF nastaje kompresijom sadržaja postojeceg drajva (za razliku od kreiranja praznog CVF od slobodnog prostora). Kada *DoubleSpace* podiza DBLSPACE.000 CVF on poziva CVF pomoću slova koje označava host drajv – najčešće C – a host drajvu dodeljuje novo slovo (*default* je H).

CVF intarna struktura podataka sadrži MDBPB (*Microsoft DoubleSpace BIOS parameter block*) Bit FAT MDFAT *boot* sektor FAT *root* direktorij i skup sektora (*sector heap*) (vidi sliku "DoubleSpace struktura podataka") MDBPB sadrži MS-DOS BPB kao i polja koja opisuju ostalak CVF a.

BitFAT označava koji sektori u skupu sektora su zauzeti (1) a koji su slobodni (0). Njegova veličina zavisi od maksimalne veličina CVF a (nalazi se u MDBPB a bira se pri samom kreiranju CVF). MS-DOS 6 iznova gradi BitFAT svaki put kada podiže CVF (i pri svakom podizanju sistema) skeniranjem MDFAT-a. Ovim procesom se verifikuje integritet MDFAT-a i BitFAT a.

MDFAT je tabele 4-bajtnih zapisa koji mapiraju svaki FAT klastar u sektore. Svaki zapis ukazuje na mesto u skupu sektora na kojem se nalaza podaci za taj klaster komprimovanu i nekomprimovanu dužinu klastera da li su podaci komprimovani ili ne i da li je klastar zauzet. Ako *DoubleSpace* ne može da kompromuje neki klastar on smešta nekomprimovane podatke da bi izbegao raspanje prostora i povećao brzinu čitanja.

Boot sektor u CVF postoji samo zbog kompatibinosti MS-DOS 6 ga ne koristi za podizanje sistema. Međutim ovaj sektor se vraća kada MS-DOS čita sektor 0 iz komprimovanog volumena.

DoubleSpace koristi standardni MS-DOS *root* direktorij (512 zapisa od po 32 bajta) i zamenjuje uobičajeni prostor za klastera normalnog FAT drajva skupom sektora. Dakle za razliku od normalnog FAT drajva, gde se prostor dodeljuje u jedinicama klastera. *DoubleSpace* dodeljuje prostor u jedinicama sektora. Kada je *DoubleSpace*-u potreban prostor na disku on dodeljuje jedan ili više slobodnih sektora iz skupa sektora i podešava odgovarajuće bitove u *BitFAT*-u tako da pokazuju da je sektor zauzet.

KOMPATIBILNOST SA FAT-om

MS-DOS fajl sistem doživljava *DoubleSpace* drajv kao normalan FAT drajv jer ima sličnu strukturu diska. Ključna razlika je u načinu na koji *DoubleSpace* skladišti podatke iz poddirektorija i klastera (vidi sliku "Kako *DoubleSpace* smešta podatke"). Na FAT drajvu, MS-DOS koristi jednostavnu formulu (sa množenjem) da bi broj FAT klastera pretvono u broj logičkog sektora na drajvu koji sadrži podatke tog klastera. Uz pomoć *DoubleSpace* drajva, MS-DOS 6 pronalazi podatke iz klastera, proveravajući MDFAT zapis koji odgovara traženom FAT klasteru.

Pretpostavimo da neka aplikacije poziva MS-DOS da bi čitala podatke iz neka datoteke. MS-DOS pronalazi FAT klaster se željanim podacima – nazovimo ga klaster X – i zatim poziva *DoubleSpace* da ga pročita. *DoubleSpace* pregleda MDFAT zapis za klaster X nalazi u skupu sektora lokaciju sektora klastera X učitava ta sektore u interni bafer vrši dekompresiju (ako je potrebna) i ponovo predaje kontrolu MS-DOS-u.

Na nekomprimovanim hard-drajvovima veličine klastera varira ali je kod *DoubleSpace*-a uvek 8KB. Prema tom broj sektora koji se čitaju iz skupa sektora može biti od jednog (ako je stepen kompresije bio 1 – 16) do 16 (ako klaster nije komprimovan).

Operacije kao što su kreiranje brisanje i traženje određene datoteke na disku na *DoubleSpace*

drajvu se obavljaju istovetno kao što bi se obavljale na FAT drajvu. Jedina razlika je u tome što čitanje poddirektorija ili klastera zahteva završavanje u MDFAT i, verovatno, dakompresiju.

Uslužne alatke za disk su već druga stvar. Svaki program koji koristi normalne MS DOS INT 21h fajl I/O operacije ili INT25h/26h operacije čitanja i zapisivanja direktno u sektore vidi *DoubleSpace* drajv kao normalan FAT drajv. Međutim, podaci za susedni niz klastera na *DoubleSpace* disku verovatno se ne smeštaju u susedne sektore u skupu sektora. To znači da će defragmentalon diska koji nisu dograđe ni da podrže MS DOS 6 i dalje raditi pouzdano... ali neće davati daloleka koja se fizički nalaze jedna pored druge. (Mnogi takvi proizvodi su već dograđeni i podržavaju *DoubleSpace*)

Disk alatke kojima raspolaže sam *DoubleSpace* CHDSK i DEFRAG, rade u dve faze kada se promena na *DoubleSpace* drajv. Prvo obave isle operacije koje bi obavili na normalnom FAT drajvu – radeć se *root* direktorijam FAT-om poddirektorijime i klastema – a zatim prelaze na operacije u jedinstvenoj unutrašnjoj struktun CVF (MDBPB BitFAT MDFAT I skup sektora).

PROCES PODIZANJA SISTEMA

DoubleSpace je integrisan u MS-DOS tako što ja kompresioni sistem DBLSPACE BIN učinjen kernel daloteksm. Pošto se *DoubleSpace* na učitava kao drajver uređuje iz CONFIG SYS nisu potrebna dve kopije daloteka CONFIG SYS i AUTOEXEC BAT (i na CVF i na nekompnmovanom disku). Takođe na morate čuvati duple kopije svih daloteka koje se navoda u CONFIG SYS i AUTOEXEC BAT (lakozvana *boot* daloteka), ni sinhronizovati promene ovih daloteka ili zamenjivati slovne oznake drajvova usred obrada CONFIG SYS datoteke.

Na sistemu bez *DoubleSpace*-a proces podiza nja počinje kada glavni mikroprocesor (CPU) izvrši ROM BIOS i učita MBR (*master boot record*). MBR traži particionu tabelu za aktivnu particiju i učitava *boot* sektor te particije. Zatim *boot* sektor učitava IO SYS i inicijalizuje MSDOS SYS. Naposlatku se izvršavaju CONFIG SYS i AUTOEXEC BAT.

U *DoubleSpace* sistemu.. DBLSPACE BIN se učitava odmah nakon inicijalizacije MSDOS SYS. DBLSPACE BIN zatim čita inicijalizacione parametre iz DBLSPACE INI i podliže CVF daloteka navedena u linij **ActivateDrive**=X: Yn. Tada nekompnmovani drajv C postaje H drajv.. dok C postaja CVF (vidi sliku "Sadržaj kompnmovanog drajva"). DBLSPACE.BIN prvo traži DBLSPACE INI na *boot* drajvu a zatim na CVF drajvu. Ako ga ne nađa DBLSPACE BIN sam napušta memoriju (*unloads*).

Pošto se DBLSPACE BIN na učitava kada nema *DoubleSpace* drajvova DBLSPACE BIN se može staviti u svaki bulabilni MS DOS 6 drajv, bez bojazni da će se uzaludno potrošiti dragocenu memoriju. A s obzirom da informacija o konfiguraciji ostaju u datoteci DBLSPACE INI na hard drajvu, možete podići MS-DOS 6 sistem sa *DoubleSpace*-om sa flopi *boot* disketa na kojoj imate DBLSPACE BIN – *DoubleSpace* drajvovi će se podići bez graška.

Sledeći korak je učitavanja *DoubleSpace* u blok gornje memorije. Kemelski proširuje ovde stvorio izvesne probleme. DBLSPACE BIN se učitava pre nego što CONFIG SYS učita sistemski upravljač 386 memorije tako da u vreme učitavanja još nisu dostupni blokovi gornje memorije. Ovaj problem smo rešili tako što smo napisali kratak program DBLSPACE SYS i njegov kod dodali u DBLSPACE.BIN tako da *DoubleSpace* može da se premasti u gornju memoriju. DBLSPACE.SYS se učitava iz CONFIG SYS nakon EMM386 EXE i HIMEM SYS i sadrži red DEVICEHIGH=C:\DOS\DBLSPACE.SYS/MOVE. Ovaj red signalizira datoteci DBLSPACE BIN da se premesti iz konvencionalne memonje u gornju memoriju.

SIGURNOST PRE SVEGA

DoubleSpace operacije možamo svrstati u one koje se ponovo stratuju (*restartable*) i robusne (*robust*). Operacija koje se ponovno startuju kao kompresija FAT drajva ili promena veličine *DoubleSpace* drajva se *moniju* završiti. Ako se takva operacija prakine pre završetka podaci na hard drajvu ostaju u nekonsistalnom stanju i računar prestaje da radi. Robusna operacija međutim kao što je CHDSK ili DEFRAG je tako projektovana da se može prekinuti bez posledica – dakle ne mora se završiti. *DoubleSpace* je predviđen za obe vrste operacija, tako da se računar može ponovno startovati u svakom trenutku bez bojazni da će se podaci izgubiti.

Kada *DoubleSpace* započne operaciju koja se more završiti, on modifikuje AUTOEXEC.BAT tako da automatski preuzima kontrolu nakon prekida. Osim toga *DoubleSpace* obavlja sve operacije na hard drajvu brižljivim redom vodeći računa da se bar jedna kopija svih korisničkih podataka nalazi na drajvu u svakom trenutku i da *DoubleSpace* u svakom trenutku zna koja operacija se obavlja. Na taj način *DoubleSpace* može naslaviti tamo gde je prekinul bez gubljenja podataka. Robusna operacija kao CHDSK i DEFRAG premeštaju podatke po *DoubleSpace* drajvu određenim redom tako da se podaci na gube ukoliko neočekivano dođe do ponovnog podizanja sistema.

PROŠIRENJE *DoubleSpace*-a

Proizvođači disk alatki mogu iskorištih *DoubleSpace* sistemski API i CVF specifikaciju da bi napisali defragmentior i druge alatke koja će moći da rada sa *DoubleSpace*-om. Proizvođači softvera i hardvara opet mogu iskoristiti MRCI (takozvani "merci" – *Microsoft Real-Time Compression Interface*), klijentsko/serverski interfejs koji omogućava kompresionom klijentu (bilo da je to *DoubleSpace* program za bekap ili transport za mrežni protokol) da koristi kompresiju bez zavisnosti od određenog kompresionog servera. Za to je potrebna samo jedna kopija kompresionog servera (kakva sreća, u ranni MS-DOS-a koja vapi za RAM-om) i odzivtava poboljšana kompresionog servera bez odgovarajuca revizije klijenta.

Najočiglednije unapređenje bi bila ugradnja MRCI servera u hardvar. To bi donelo nekoliko prednosti: RAM koji zauzima softverski server više ne bi bio potreban i bio bi odslupan drugim programima (sadašnji MRCI server zauzima 14 KB). Hardverski server je brži i može postići bolju kompresiju jer mnogo lemeljitije traži podudarnosti. U multitasking okruženju hardverski server može osloboditi glavni mikroprocesor za druge funkcija.

Hardver na kojam buda primenjen MRCI će naj veću brzinu postizati na arhitekturi sa lokalnom magistralom. U budućnosti bi MRCI kompresija čak mogla biti ugrađena u CPU. Sa druge strane kompresiona kartica se ISA magistralom nije u povoljnom položaju pošto njena brzina od 8 MHz ograničava količinu podataka koji se mogu preneti sa i do kartice. MRCI je trenutno ograničen na rad u stvarnom režimu, ali će uskoro podržavati i *enhanced Windows* režim.

Pri projektovanju *MemMaker*-a i *DoubleSpace*-a, u središte pažnje smo stavili sigurnost u nadi da će mnogo šira publika imati koristi od povećana konvencionalne memorija i prostora na disku. Do sada je na te blagodeti imao pravo samo uski krug posvećenih naprednih korsnika.

Bendžamin V. Slivka je menadžer razvoja. Enk Strob glavni programski menadzar a Ričard Fridman glavni proizvodni menadžer za MS-DOS 6 u "Microsoft"-u (Redmorid Washington). Možete ih naći na BIX-u kao *editors* ili na "Internet" u kao *editors@bix.com*

Nastavak sa strane 57

nog sistema treba da uzmu u obzir vreme potrebno ili onemoguće neki prekidi su instrukcije CLI i STI koje se odnose isključivo na INTR liniju mikroprocesora tj. na maskirane interapte. Ipak u radu se može dogoditi da neki interapti budu maskirani nezavisno od uticaja programera, uglavnom iz razloga što pnontetnji interapti već „čekaju na red". Najviši pnontet dodeljen je softverskim prekidima koji nisu vezani za debagar (npr *page fault*). Sledeći su softverski interapti odnosno *trap* instrukcije (npr. INT 0 ili bopšte INT *n*). Sliede debagerski prekid za tekuću instrukciju i debagerski prekid za sledeću instrukciju nemaskirani interapt (NMI: gde su ona vremena kada je NMI bio interapt najvišeg pnonteta?) i najzad INTR. Ako tokom izvršavanja jedne instrukcije nastupe *page fault* i *debug fault*, najpre će biti izvršen prvi kako bi se potrebna stranica odvela u memoriju a tek onda se dopušta *debug fault* kako bi se proverilo dejstvo instrukcije – da je obmuto instrukcija uopšte ne bi mogla da se izvrši (nema podatka) pa bi i debagovanje bilo bespredmetno!

Ostala intarna maskiranja prekida nastupaju kada se primeti NMI (dalji nemaskirani prekidi su onemogućem do izvršavanja instrukcije IRET) kada na stupi *debug fault* (RF bit registra EFLAGS se setuje čima su dalji prekidi debagera onemogućeni! RF će sam procesor obrisati kada završi se tekućom instrukcijom) i što je posebno važno kada se menja vrednost registra SS (stek segment). Tokom izvršavanje la i sledeće instrukcije nemaskirani i maskirani za zamenu posiova koje je značajno veće od vremena potrebnog za pozv „običnog" interapta koji će delu prostor konsnikove aplikacije – u praksi će samo naki interapti (npr. pomenuti *double fault* i *invalid TSS*) imati svoj TSS dok će za ostale sve ići po sta rom uključujući i obavezu konsnika da predvidi potreban pmstor na steku ili barem da omogući njegov rast preme nižim adresame.

Format ulaza IDT a koji se odnose na hardverske prekide i *trap*-ove ja identičan i prikazan na slici 3 – jedina razlika je polje TYPE. Uostalom, i izvršavanje ove dve vrste prekida je gotovo identično s tim što hardverski prekid postavlja IF bit EFLAGS registra na nulu. Time su dalji hardverski prekidi onemogućeni do kraja obrade tekućeg ili bar od trenutka u kome program koji obrađuje interapt na izvrši Inetrukciju STI.

Rutina za ob`..uu prekida završava se instrukcijom IRET koja vraća originalnu vrednost brojača naredbi flegove i stek segment. Ako je NT (*nested task*) bit EFLAGS registra setovan sledi i zamena poslova na osnovu originalnog TSS-a, ako interapt nije posao za sebe pn pisanju interapt procedura se mora strogo pazrti da ne staku na celanu neki podeci koji bi poremetili rad prekinutog programe. Čitav meharizam ilustruje slika 4.

PRIORITETNI INTERAPTI

Što se programera tiče, jedini način da se spreče prekidi su automatski onemogućeni kako bi se omogućilo konsniku da definiše i sadržaj ESP. *Intel*, na žalost ovde nije doteran stvari od kraja pošto „između" punjenja SS-a i ESP-a nisu onemogućeni softverski interapti – autor ovoga teksta imao je slične probleme sa jednom aplikacijom koja se „rušila" potpuno nepredviđeno i to po osnovu *double fault*-a. Najzad smo ustanovili da se ponekad dogodi *page fault* „između" instrukcija MOV SS, i MOV ESP pa rutina koja ga obraduje zatekne potpuno nedefinisani stek. Greška je otklonjena konšćenjem instrukcije LSS koja „odjednom" puni SS i ESP. Naravoučenja ne izbegavajte složene instrukcije na mogu se uvek zameni ti nizom jednostavnijih!

U sledećim „Računanma" detaljno ćemo proučiti sve 80386/80486 prekida i upoznati se se mogućnostima debagovanja programa i samog operativnog sistema.

OLYMP electronic

11000 BEOGRAD
Jovana Đaje 10
tel. 011/400-477
fax 011/410-240

EPSON
EIZO
Roland DIGITAL GROUP

386-33 SX

RAM 1Mb
FLOPPY 5,25"
HD 42Mb
IDE CONTROLER 16bit
VGA 512Kb
SVGA MONO MONITOR 14"
DESKTOP CASE
TASTATURA 101

1180

386-40 DX

RAM 4Mb, 64Kb CACHE
FLOPPY 5,25"
HD 85Mb
IDE CONTROLER 16bit
SVGA 1Mb
SVGA MONO MONITOR 14"
MINI TOWER CASE
TASTATURA 101

1890

486-33 LOCAL BUS

RAM 4Mb, 256Kb CACHE
FLOPPY 5,25"
HD 120Mb
IDE CONTROLER 16bit
SVGA 1Mb
SVGA COLOR MONITOR 14"
MINI TOWER CASE
TASTATURA 101, MIŠ

2990

486-50 LOCAL BUS

RAM 4Mb, 512Kb CACHE
FLOPPY 5,25"
HD 210Mb
VL-BUS CONTROLER 32bit
SVGA 1Mb VL-B 16,7 mil. boja
SVGA COLOR MONITOR 14"
MINI TOWER CASE
TASTATURA 101, MIŠ

3590

486-66 EISA

RAM 8Mb, 512Kb CACHE
FLOPPY 5,25" i 3,5"
HD 540Mb SCSI
SCSI EISA CONTR. 4Mb
SVGA 1Mb 16,7 mil. boja
SVGA COLOR MONITOR 14"
LARGE TOWER CASE
TASTATURA 101, MIŠ

8190

Mainboard

MB-386/33 SX	210
MB-386/40 DX	370
MB-486/33 LOCAL BUS	1000
MB-486/50 LOCAL BUS	1350
MB-486/66 EISA	1850

SVGA

SVGA mono monitor	250
SVGA color monitor	650
VGA 512Kb	110
SVGA 1Mb	150
SVGA VL-BUS 16,7 mil. boja	320

Diskovi

HD- 42Mb	350
HD- 85Mb	430
HD-120Mb	500
HD-210Mb	680
HD-320Mb	980

Komponente

Floppy disk 5,25"	150
Floppy disk 3,5"	130
Memorija SIMM 1Mb	150
16bit IDE controler	50
32bit VL-BUS controler	150

Dodatna oprema

Ethernet card 16 bit	220
Miš	50
Filter stakleni /14-16/"	140
Ribon za A4/A3	20/30
Centronics kabl	30

Kućišta

DeskTOP	160
Slim	200
Mini TOWER	200
TOWER	240
Large TOWER	300

EPSON

Printeri

LX-400	470
FX-1170	1190
LQ-570	860
LQ-1070	1250
LQ-1170	1650

EIZO

Monitori

6500	21"m	2900
F340I	15"	1950
F550I	17"	2750
T560I	17"	CALL
T660I	20"	CALL

Roland

Ploteri

DXY-1100	A3	1780
DXY-1200	A3	2210
DPX-2500	A2	CALL
GRX-300	A1	CALL
GSX-4000	A0	CALL

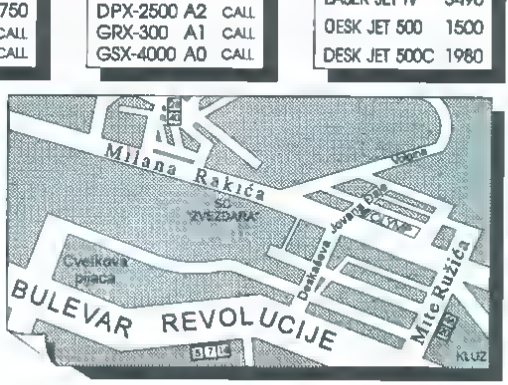
HP

Printeri

LASER JET IIIP	2350
LASER JET IV	3490
DESK JET 500	1500
DESK JET 500C	1980

PONUDA ZA DILERE

EPSON EPSON
EPSON EPSON
EPSON EPSON
EPSON EPSON
EPSON EPSON



TECO 21000 Novi Sad, Čirila i Metodija 166
tel. (021)/392-975, fax (021)/395-149

INTELIGENTNI KONTROLERI

U prvom delu ove mini-serije, razmatrali smo razloge za i protiv keširanja diska, kao i RAID tehnologiju. Ovaj nastavku ćemo posvetiti razmatranju raznih vrsta keš kontrolera u kontekstu veze sa računarom.

Dejan V. Veselinović

U prvom nastavku smo opisali principa rada inteligentnih keš kontrolera kao i osnovnu prednost koju oni nude u poređenju sa raznim programima za keširanje tvrdog diska. Sada nam je preostalo da istražimo još dva aspekta ove teme, a to su vrsta veze sa računarom i uticaj tih veza na performanse i kombinovanje keš kontrolera i programa za keširanje (sprega mašinskog i programskog keširanja)

VRSTE VEZA

Kao i u slučaju svih ostalih veza i kartica i za keš kontrolere prema okvirnom IBM AT 339 standardu postoje tri osnovne veze u slučaju ličnih računara obična ISA sabirnica EISA sabirnica i „Local Bus" (LB) sabirnica Isti na tu je i sabirnica koja koristi IBM mikrokanal, ali praksa pokazuje da se radi o relativno egzotičnom standardu bar za naše podneblje, pa se ovom vezom necemo baviti

ISA sabirnica koristi IBM AT 339 standard u svakom smislu radni takt joj je 8 MHz sa jednim stanjem čekanja a pretpostavljena veza sa ostanja na ST506 standard Tipične brzine prenosa koje se postižu sa ovom vezom iznose 1 9 2 9 MB/s za čitanje i 1 8-2 7 MB/s za pisanje u spoljnoj memoriji Rasponi se donose na podešavanje parametara rada sabirnice što se sasvim lako obavlje sa novijim verzijama programa za podešavanje (SETUP u BIOS-u) posebno sa AMI i MR BIOS-ima

EISA sabirnica nudi bar prema nominalnim standardima brzine protoka podataka i do 33 MB/s na žalost ovakve brzine su u praksi veoma retke Za poslednje tri godine ni jedan jedini put nismo videli ovaj standard izveden do kraja donosno do ovih nivoa Ove brzine bi trebalo da se postižu pre svega dupliranjem širine sabirnice sa 16 na 32 bita Druga velika prednost EISA sabirnice je mogućnosi tzv „bus mastering" efekta, po kome, pojednostavljeno rečano neki uređaj može u nekom momentu da preuzme kontrolu nad celom sabirnicom obavi zadati posao i nekon toga vrati kontrolu drugim uređajima

LB sabirnica u neku ruku predstavlje pokušaj da se i sama sabirnica automatski prilagodi radnom taktu cele mašine lako se i ona ponaša prema nekim okvirnim standardima (OPTI li češće VESA), ona je 32-bitnom putanjom veoma neposredno vezana za sam procesor pa kada on promeni takt, menja ga i ona u donosu 1 1 To znači da ako, recimo zamenite procesor 486DX-25 istim na 33 MHz, i veša sabirnica će automatski prihvetiti ovo prepodešavanje procesora Teorijski sve ća se automatski i bez problema dovijeti do takta od 50 MHz

Dakle ovo su tri verzije keš kontrolera koje smo testirali SCSI veze je na žalost ostala za neki drugi put mahom zbog toga što smo želeli da i u tom slučaju možemo da isprobamo i LB verziju Ie veze a to u trenutku pisnja teksta nije bilo moguce Šteta, ali ima dana

KONTROLERI

U osnovi sve tri kontrolere nude potpuno iste mogućnosti i očigledno poiču iz istog izvora Sva tri mogu da povežu do četiri IDE diska kapaciteta do 4 GB po disku sva tri se mogu lako ukomponovati u Unix i Novell upravljačke sistema Sva tri podržavaju do 18 MB memorije u vidu četri SIM modula sva tri imaju istovetne programe i opcije za podešavanje koji se nalaze na EEPROM-u na pločicama Najzad sva tri kontrolera podržavaju do dve flopi disk jedinice na žalost, samo do kapaciteta od 1,44 MB (još uvek na i do 2 88 MB što bi bilo zgodno)

Verzije za ISA i EISA sabirnice koriste procesor Intel tAPX88C186 na 12 MHz ako je ovo normalno za ISA sebirnicu malo nas je začudilo u slučaju EISA sabirnice Naime neki proizvođači kao recimo MYLEX koriste čak Intal tAPX80386SX 20 procesore malo je čudno da se na 32-bitnoj sabirnici koristi hibridni 16/8 bitni procesor uprkos činjenici da kontroler sa diskom obavlja sve poslove posredstvom 8-bitne sabirnice saglasno ST506 standardu

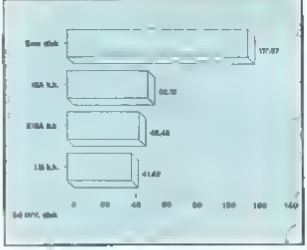
Međutim verzija za LB koristi procesor Intel tAPX80C186 donosno punu 16/16-bitnu verziju ovog procesora takođe na 12 MHz To je već mnogo bolje pa su i naša očekivanje porasle Naravno usled pune 16-bitne sabirnice na ovaj kontroler se moreju staviti najmanje dve SIM mdoule dojednom

Prema deklaraciji ISA kontroler razmenjuje podatke brzinom do 5 MB/s i sa sabirnicom i sa diskom EISA verzije razmenjuje podatke sa diskom brzinom do 5 MB/s ali sa sabirnicom radi brzinom do 9 MB/s trajno donosno do 16 5 MB/s vršno LB verzija sa diskom i dalje komunicira brzinom do 5 MB/s ali sa sabirnicom radi brzinom do 20 MB/s vršno donosno 14/19 MB/s trajno za Iekt procesora/sebirnice do 33/50 MHz

Sva tri kontrolera su bila opremljena sa po 4 MB memorije Odabrani disk za ovaj test je bio Fujitsu mdoel MT2614E, disk ima kapacitet do 160 MB formatirano brzinu slučajnog pristupa od 18 mS brzinu rednog pristupa od 10 2 mS Povrh svega kriva brzine prenosa ovog diske je za današnje pojmove čak dosta loša Ovome se ne treba čuditi pošto sa radi o stanjem modelu što se najbolje vidi po rednom pristupu ovo je valjda bio poslednji takav disk Ikada napravljen Namerno je doabran upravo zbog tih mena i to iz dve razloge prvo ako iše treba ubrzavati onda je to disk sa lošijim performansema i drugo relativno loša performanse diske najbolje ilustruju dobiti od kesiranje

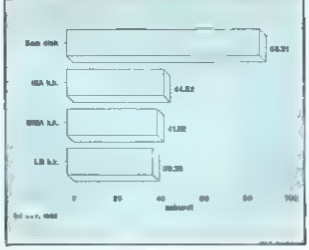
Da se radilo o nekom boljem disku recimo Quenjum modelu LPS240A koji umeselo 16 KB lokalnog keša ne Fujitsu disku ima 256 KB vlsesegmentne keš memorije neki rezuitati testova bi bili mnogo bolji Podsećamo da kontroleri MOGU da razmenjuju podatke sa diskom brzinom od najviše 5 MB/s međutim mi do sada nismo videli ni jedan jedini IDE disk koji je značajnije prebacivao polovinu Ie brzine Ovde se radi o faktoru brzine prenosa tako se keš kontroler nalazi između procesora i diske Ipak je apsolutna brzina rada i Ie kako pod uticajem brzine razmene podataka između diska i kontrolera. Na kraju krajeva to sa lako može

Slika 1



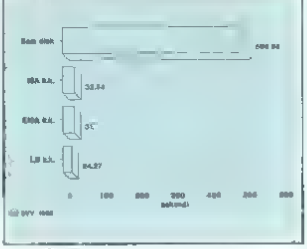
Preiirazivanje baze podataka, 981 Kb

Slika 2



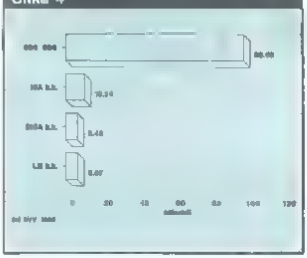
Računanje baze podataka, 981 Kb

Slika 3



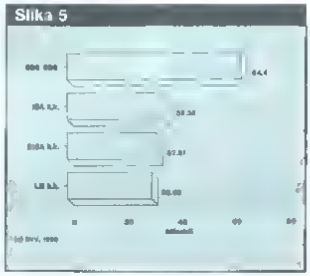
Indeksiranje baze podataka 981 Kb

Slika 4



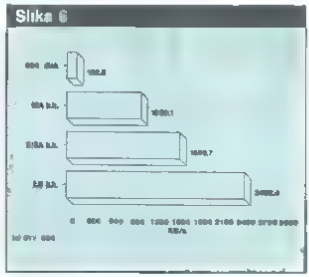
Indeksno pretraživanje baze podataka, 981 Kb

Slika 5



Sortiranje baza podataka, 981 Kb

Slika 6



Male datotake, 200 Kb

videti prilikom testiranja brzine pranosa sa kontrolera u sistemsku memoriju prostom zamenom diskova razlika je naravno mnogo manja nego kada sa porede diskovi sa običnim kontrolerima ali i dalje postoji i nije zanemarljiva jer se kreće u rasponu 10-30% sa kontrolerom u odnosu na 1 3 bez kontrolera (dakle ne to razlika pomarena za jedan red veličine ma da napominjemo da je ovaj zaključak diskutabilan kao nako opšte pravilo)

Svi kontroleri su bili podešeni na veličinu reči od 16 kB čitanje unapred od 3 saktora, BIOS-om preseljnim u RAM u režimu „write back" rada i sa vremenom pisanja odradenim kao „beskonačno drugim rečima, bili su podešani na maksimum brzine rada Matična ploča je koristila procesor Intel iAPX80486 na 33 MHz sa 256 KB keš memorije i 4 MB sistemske memorije u svim slučajevima Koristili smo jednu ploču koja na sebi sadrži i ISA i EISA i LB utičnice

REZULTATI MERENJA

Bitni rezultati su prikazani na slikama 1-8 Slika 1 prikazuje brzinu pranosa bloka od 64 KB u Irajanju od 10 sekundi Lako je uočiti da sa razlike između kas kontrolera kreću u odnosu 1 7 3 a indeks performansi najboljeg kontrolara u odnosu 1 22 9 što već samo po sebi govori o razlikama u performansama

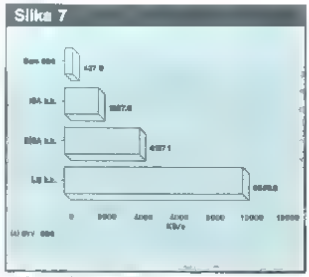
Slika 2 prikazuje rad sa malim datotekama od po 200 bajtova Primatićemo da se indeks performansi ISA kontrolera u odnosu na sam disk veoma bitno popravio sa 7 3 1 na 2 4 1 Slika 3 prikazuje razultata leslova sa velikim datotekama od po 2 KB Razlika u performansama između kontrolera su se povećala na 5 3 1 a razlika u performansama između samog diska i najbrzeg kontrolara je zdravih 1 23

Međutim sva ovo su laboratorijski leslovi Najinteresantnija je slika 4 koja pokazuje ukupna prosečne indeksne performansi na patu leslova sa realnim bazama podataka (Slike 5 10 pokazuju razultata na pojedinačnim testovima u stvarnom vremenu tj u sekundama potrebnim da se obavi posao)

Primaćemo da ja ukupna stvarna razlika u performansama bez i sa kontrolerom kudikamo manja nego u slučaju laboratorijskih leslova U najgorem slučaju ona iznosi 1 6 24 što je dobar razultat međutim daleko je interesantnije primetiti razliku između najboljeg i najgoreg kontrolara Od međusobno veoma velikih razlika u laboratorijskim leslovima u stvarnosti razlika sa avala na jedva 36% Objašnjenje je u tome što se u stvarnosti procesi rada odvijaju sa manje ili više slozanjim skupovima naredbi nego u laboratorijskim leslovima ovo ističemo zbog vernika o proste laboratorijske leslova – uvek pa i sada napominjemo da oni mogu biti vrlo indikativni ali nikada ne treba da budu apsolutizovani kao kriterijum izbora jer su najčesce previše pojednostavljeni

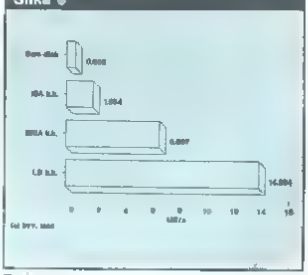
Ocigledno razlike u performansama između ISA i EISA kontrolera koji su inače identični jasno ukazuju na činjenicu da je brzina pranosa podataka sa kontrolara u sistemsku memoriju ključni elemanat brzina rada. Slučaj LB kontrolera koji ima jos veću brzinu pranosa, uz duplo širu vezu procesora na kontroleru sa sistemskom memorijom (16 umesto 8 bita) samo potvrđuje gornji zaključak

Slika 7



Velike datoteke, 200 Kb

Slika 8



Brzina prenosa

To nas dovodi do kombinacije mašinskog i programskog keširanja Probe radi upotrebili smo keš program iz popularnog paketa „PC Tools", ali provere radi i program „PC Kwik" Sasvim očekivano i sa običnim ISA kontrolerom dobili smo višestruko bolja rezultate u svim testovima Svi rezultati su se popravili za faktor 3 ili vise čima je skromni ISA kontroler pravazišao rodaka namenjenog lokalnoj sabirnici Ostaje nam samo da ponovimo saval od ranije – iako imale inteligentni keš kontroler opet ostavile makar i skromnu količinu sistemske memorija za kesiranje Dobicete brzinu gotovo nalik na brzinu rada sa RAM diskom uz sve prednosti kapaciteta, bezbednosti i mogućnosti koja vam ni jedan RAM disk ne može pružiti

ZAKLJUČAK

Na kraju ostaje nam da razmotrimo neka odnosa između cena i performansi Ako ISA kontrolar košta 1 EISA verzija košta 1 68 a LB kontrolar košta 1 1 (!) Ovo je naizgled apsurd – sponji košta viša od brzeg a najbrzi je samo 10% skuplji od najsponjeg Međutim nema tu misterija

LB kontroler jesla najbrzi ali je u osnovi sasvim isti kao i najskromniji razlika je jedino u vezi sa procesorom samom procesoru na ploči (koji uzgred dodaje samo USD 1 po ceni) i malo koda u BIOS-u kartice što sva zajedno i vredi oko 10% U slučaju EISA kontrolera bilo je potrebno dodati i kompletnu logiku za EISA identifikaciju kao i logiku za „bus masteing" što je objaktivno komplikovanije i teže pa samim tim i mora da košta više

Ovi odnosi cena jasno ukazuju kojim putem treba poći Ako sada po prvi put kupujete računar ili uzimate novu mašinu svakako se opredelite za LB verziju, jer je razlika u ceni potpuno zanemariva u odnosu na dobiti u performansama (1% doplate vam donosi 3 9% performansi tj dobijate 4 puta više u odnosu na doplatu)

Ako već imate računar sa svim potrebnim u njemu a nemate LB opredelite se za ISA varziju jer konacno i nemate izbora sem opsežnije zamene dalova

Ako ste vlasnik EISA mašine onda ste pred dosta teškom dilemom da li da se opredelite za jeftiniju ISA varziju kao rešenje sa boljim odnosom cena/performansa ili za EISA varziju koja je objaktivno tehnički savršenija i nudi mogućnosti koje ISA kontroler po definicip nama („bus masteing" pra svega) Ne bismo voleli da se nademo na vašem mestu sa ovom dilemom

DA LI JE NEKO VIDEO TANJU?

U obradi teksta ni izdeleka nija rečena poslednja reč. Šte se tranutno događe u ovoj oblaeti u svetu, i kakvog uticaja na sva to ima, prako svoje jaka američke veze, jedna zagonetne Beograđenke po imenu Tanje?

– *Kažete imate tu na disku tekst datoteku od 600 MB?*
– *Da*
– *I možete da je pretražujete?*
– *Aha i ne samo to, mogu da zadam komandu poput „pronađi sva mesta gde se javljaju reči VLADA KOSTIĆ i PROGRAM, pod uslovom da je u blizini reč IBM PC ali ne i reč ZX SPECTRUM"*
– *He he, sa 600 MB to ima da traje večnost i malo duže*
– *2-3 sekunde, na ovoj 386-ici*
– *A ???!!!*

Dobrodošli u svet obrade teksta! Stvar je počela pre jedno dvadesetak godine, kada je firma „Infodata" razvila prugremčić *Inquire* za „text storage search and retrieval". Neravno, stvar je radila na IBM-ovim dinosaurusima. Prizanajem da „infodate" možda nije prva napravita takav program, ali šta me bnga – morem da hvalim svoju firmu. Dakle, živela „Infodata"!

OK program služi da se bilo kakev tekst smesti na disk, a zatim pretražuje po volji i delovi prikazuju na terminalu. Lepo a ú čemu se lo razlikuje od baze podataka? Baze podataka manipulišе informacijama pomoću slogova i to obično slogove fiksne dužine po mogućnosti na suviše velikim, i fiksnim polima unutar sloga. *Inquira* i slični programi manipulišu tekstom (kao što je ovaj samo mnogo dužimj koji naravno na može da se podeli na slogove fiksne dužine. To je jednostavno niz tekstova (raznorazni dokumenti, časopisi, novinski izveštaji itd, itd, itd) lepo skupljenih na jednu gomilu dužina koje se meri megabajtima, eko ne giga bajtima.

Za konsnika to, naravno, nije nepregledna gomile teksta. Konsnik može da pogleda sadržaj, a može i da zada vrlo složene komande, takozvene „Queries" u stilu „pronađi mi ovo i ono pod ovim i onim uslovom, i tako to". Posle toga, program ga izvesti koliko ima „pogodaka", i on (voljeni korisnik) može da pogleda neke od tih pogodaka, odštampa delove koji ga zanimaju i radi šta hoće.

Milina prave – ali koma treba tško nešto? Svim papirolozima, naročito onim sertifikovanim, kao što su državne institucije, vojska i velika firme – banke, trgovina, sudstvo, itd. Možete na disku da imate sve zakone i propise, svu korespondenciju cele firme, izveštaje, tehničku dokumentaciju i još sto čuda.

„Infodate" u stvari ima samo dve vrste klijenata: vladine institucije i velike firme – praktično sve firme na spisku 500 najvećih u Americi. Da, vrli se fu velliki novac.

NOVI TALAS

Poput dinosaurusa ne kojima se izvršava, *Inquire* je i sam dinosaurus nenormalne dužina, pisan i krpljen tokom vremena na Cobol-u, Fortran-u, PL-1 i asembleru. „Spaghetti code", tako sa to zove.

„Spaghetti code" danaa više nije u modi. Sada su u modi lepo strukturani programčići, pisani ne C-u ili C++. Ni IBM-ovi dinosaurusi više nisu aktuelni.

Vlada Kostić

Kao posledica toga javio se ceo novi talas programa za obradu teksta: sva do jednog zasnovani na „client-server" arhitektur. *Open Text, Excalibur, Fulcrum, Mega Text, Conquest*, i možda još nekoliko drugih za koje je ne znam.

Za „client-server" arhitekturu, bitno je pre svega imeti dobru mrežu. Dakle, počnemo od jedne 486-ice na koju instaliramo *Novell Netware 3.11* ili *4.0*, i dodamo gomilu 386/486 mašina na kojima se vrti DOS/*Windows*. Ne to sve dodamo jednu *Unix* mašinu (može da bude 486-ica, SUN, IBM RS-6000, Hewlett Packard, itd).

Na tu mašinu nekačimo ogroman hard diek (ili optički *read/write* disk) i dobili smo naš tekst-server. Seda tu instaliramo *Excalibur, Open Text Maga Text* ili nešto slično i pola posla je gotovo. Klijenti su, naravno 386/486 mašina u mreži. Softver za klijenie obično radi pod *Windows*-om (to je postale velika mode) i obavlja sledeće funkcije:

- Omogućava konsniku da konstruiše „query"
- Šalje taj „query" serveru
- Prima odgovor od servera o broju pogodaka
- Omogućava konsniku da pogleda te pogodke

Softver na serveru obavlja sledeće funkcije

- Skladišti teksi datoteku
- Indeksira datoteku
- Pretražuje datoteku kada dobije „query" od kljenta.
- Šalje klijentu broj i lokaciju pogodaka
- Šalje klijentu delove teksi datoteka (to su obično stranice na kojima se nálaze pogoci i koje korisnik želi da pročita)

To je ono što je više-manje slično za sve sisteme. Malo detaljniji pregled pokazuje mnoge razlika: kako u pogledu funkcionalnosti tako i po performansama.

Kada smo već kod performansi: mislite da ću da odem tajnu kako se pretražuje 600 MB za 3 sekunde na 386-ici? Žalim, ali neću – nemam pojma keko se to radi. Kao dobro plaćeni konsultant, moj posao je da drugima solim pamet na da izgrevam programere i pišem algoritme.

Međutim, znam da je potrebno napraviti „indeks" datoteke. To naravno ima smisle – samo čitanje 600 MB sa diske bi trajalo dobrih 10 minuta, a možda i mnogo duže eko je datoteka na optičkom disku. Najpopularniji metod da se napravi indeks je invertovana lista: mada se često konste i binarne liste. Firma „Excalibur" je otišle korak dalje i koristi „neuronske mreže".

INDEKSIRAJ ME

E sada postoje indeksi i indeksi. *Inquira* pravi indeks višestruko duži od originalne datoteke. *Open Text* indeks iznosi 70% originalne datoteke, dok se *Excalibur* spustio ne samo 30%. I ne samo to – eko je verovati propagandi *Excalibur*-ovih stručnjaka, njihovo remek-delo je u slanju da pretražuje datoteku konsultujući samo indeks – bez potrebe da čita originalni teksi. To je vrlo važno ako se originalne datoteka nalazi na optičnom disku, koji baš nije šampion brzine. Naravno, indeks (pošto je mnogo manji) može da stoji na hard disku. *Open Text* sa druge strane, pretražuje datoteku čitajući indeks a kada zatreba, i originalnu datoteku. To radi oko 5 do 10 puta pre nego što pronađe sve pogotke u datoteci. Međutim *Open Text* ima jednu drugu prednost: ako je verovati njihovoj propegandi, vreme pretrage skoro ni malo ne raste sa porastom datoteke. Ali ne lezi vraže – sada *Excalibur* tvrdi da njihove neuronske mreže, pod određenim okolnostima, u stvari rade sve brže i brže kako datoteka raste. Gle čude! Ne misle valjde da im verujem?

Tu nije kraj priče o indeksima. Kada se doda parče teksta na datoteku, neki sistemi indeksiraju celu datoteku iz početka – a to zna da potraje i danima. Drugi indeksiraju samo deo koji se dodaje, što je već mnogo bolje. Međutim, ni jedan sistem koji sam video do sada ne ume da indeksira datoteku u realnom vremenu – recimo u situaciji kada bismo svakih nekoliko sekundi dodavali po nekoliko redove takste datoteci.

PRONAĐI ME

Indeks je samo pola posla. Druga polovina je upotrebiti taj indeks da bi se pronašlo nešto u datoteci. Naravno da je besmisleno imati najobičniju *search* funkciju (kao recimo F2 kod *WordPerfect-a*) kada imamo datoteku dugačku gigabajt ili više. Često konsnik ne zna ni šta sve tu ima, a još češće nije siguran ni šta tačno želi da pronađe. Zato postoje razni uslovi, džoker i „fuzzy logic search".

Uslovi mogu da budu uobičajeni „AND" i „OR", a postoji i „proximity search" – kada jedna reč treba da je u blizini neke druge reči. Delje pomoću „NOT" moguće je pronaći sve meste gde se javlje jedna reč ali ne i neka druga reč u istom pasusu – iako smo došli do onoga „pronađi reči VLADA KOŠTIĆ i PROGRAM, pod uslovom da je u blizini reč IBM PC ali ne i reč ZX SPECTRUM". Neki sistemi, *Open Text* na primer, imaju kompletan mali programski jezik samo za pretraživanje teksta. Džokeri rade isto kao pod DOS-om. Komanda „pronađi TANJ*" će pronaći „TANJA", „TANJICA", „TANJICE", i tome slično. „Fuzzy logic" stvar podliže na sosvim drugi nivo. Komada „pronađi TANJA VOLIM TE JAKO PUNO" će smatrati za pogodak ne samo „TANJICE VOLIM TE JAKO PUNO" nego i „TANJA PUNO TE VOLIM" možda čak i „TANJICE LJUBAVI, VOLIM TE". U stvari pogodak bi čak bio „VOLIM TE JAKO PUNO TAMARA" ali to se Tanjici ne bi baš dopalo. To su le „neuronske mreže" na delu i stvar divno radi.

Preciznost se može podesiti od 1 do 100%. Preciznosi od 100% će lražili tačno ono što se zada i ništa drugo. Preciznosi od 1% će sve i

svašta smetrati pogotkom *Excalibur* je jedini sistem koji podržava „fuzzy logic search"

„Fuzzy logic search" ima takođe radostan efekat na pravopis To za nas nije mnogo bitno hvala Vuku Karadžiću, ali Englezi nisu bili prosvećeni na isti način Ako zadate programu da pronađe „TANYA NOW AND FOREVER", a to se pravilno speluje „FOREVER" neravno da ćete doći nula pogodaka Tu će *Excalibur*-ovi predstavnici da skoče na noge i kažu kako je njihov sistem jedini koji će hvala neuronskim mrežeme da vam garantuje pogodak Tipografskih grešake nismo ni mi pošteđeni Ural još jednom za *Excalibur*

Neke firme sada rade na još naprednijim sistemima uz pomoć veštačke inteligencije Takavi sistemi će biti (ili već jesu) u stanju da odgovore na pitanje! Recimo, otkucamo „Da li volim Tanju?" program protrči kroz datoteku, pronađe izjavu „TANJA VOLIM TE JAKO PUNO" i odgovori „Da Vlada voli Tanjicu!" Divno eh eko mu sada zadamo pitanje „A da li se oni grle i ljube?", glupa kanta neće odgovoriti „Verovatno"... zato što ne zna de to zeljubljeni obično rade Čista budalaština Ako mene pitate, veštačka „inteligencija" još dugo neće postati inteligentna

PRIKAŽI ME

Sledeći korak je prikazati pogotke na ekranu Većina sistema će konsniku pokazati celu stranicu na kojoj se nalazi pogodak (ili pogoci, ako ih ima više). Konsnik to može lepo da pročita, a može da ide stranicu napred ili nazad po želji Može naravno, de redi i druge stvari – recimo, poglede sadržaj i onda direktno ode na deo koji ga zanima

Prikazati stranicu na ekranu nije lako Vrsme tekst ekrana 25x80 je odavno prošlo Danas se koristi *Windows* i grafika visoke rezolucije To znači, mogući su razni fontovi grafike i još sto čude Sve je to lepo ali postoji i problem nama univerzalnog standarda za obeležavanje teksta!

Program koji prikazuje stranicu teksta na ekranu se zove „viewer" *Viewer* mora da je dovoljano inteligentan da formatira i prikaže tekst kako treba. Recimo, ako sekvenca CHR(4) CHR(22) CHR(1) predstavlja „bold" tekst onda to ne ekranu treba de se pojavi kao „bold" i nikako drukčije

Standarda za obeležavanje teksta ima meli milion pa svi sistemi o kojima ovde govorimo imaju „viewer" kao zasebon modul. Evo vam *viewer* za ovaj i onaj standard, a ako baš želite nešto specijalno evo vam i tehnička dokumentacija, pa napišite sami svoj *viewer*

Suočena sa tim problemom američke vlade je za svoje potrebe definisale standard koji se zove SGML (*Standard Generalized Markup Language)* SGML konsti komande u uglastim zagradama – tako se recimo početak naslova obeležava sa <TITLE> kraj naslova sa </TITLE>, početak *bold* sekvence se <BOLD>, kraj sa </BOLD>, i tako dalje SGML je u poslednje vreme postao izvanredno popularan i van amerčke vlade, ali nisu svi sistemi SGML kompatibilni – bar na za sada.

NEMOJ DA ME ZBUNIŠ

Viewer ze SGML je samo deo problema Drugi deo je program za indeksiranje Recimo, ovo je razgovor koji sam vodio sa predstavnicima firme *Excalibur*

– *Već sistem podržava SGML?*
– *Da,* naravno *možete konstiti viewer koji želite, uključujući i nekoliko raznih SGML viewer-a*
– *Lepo e da li će SGML komada da zbune program za indeksiranje?*
– *Naravno da neće sve u originalnoj datoteci što nije ASCII (karakteri od 32 do 127) naš program za indeksiranje jednostavno ignoriše*
– *Da, ali SGML komande jesu čist ASCIII*
– *Ovaj..., pa..., u tom slučaju ..., a je l' ste si gumi?*

Eh da, tu je problem Konsnik, recimo, otkuca komandu „pronađi mi TITLE" Program pronađe hiljadu mesta gde se javlja <TITLE>, prijavi hiljadu pogodaka, korisnik zatim želi da vidi neki od tih pogodaka – *viewer* sekvencu <TITLE> naravno neće prikazati nego će lepo preći na krupnim slovima zato što naslov valja štampati krupnim slovima I sada ne ekranu nigde reči „TITLE", a program je prijavio pogodak

Open Text sa druge stane neće pasti na tom ispitu program za indeksiranje jednostavno pročita datoteku u kojoj je definisano sve što treba ignorisati

I PONOVO INDEKSIRANJE

Pravilno obeležavanje teksta nije važno samo za *viewer* Ako je tekst lepo obeležen, program za indeksiranje može de automatski generisne sadržaj U protivnom, ima de sa ubijeme redeći to ručno

Za sadržaj je bitno imati <TITLE> i </TITLE> svugde gde treba Međutim, ako imamo I <AUTHOR> </AUTHOR>, onda ćemo pored sadržaja imati I listu autora Sa <SUBJECT> </SUBJECT> možemo da dobijemo listu teme, i tako dalje

A šta ako je originalni dokument u *WordPerfect* formatu a ne SGML? Postoji program koji će *WordPerfect* format da konvertuje u SGML, ali tu nije kraj poše Većina stvari koje treba da su obeležena verovatno tu nisu (Marfijevi zakoni) Sledeći korak je neki od SGML editora i manuelno obeležavanja E maleeeeral U takvoj situacije se tekst lepo pošalje u Venecuelu, Dominikansku Republiku, Ili na Filipine i posao bude urađen po lokalnoj tarifi od 0 0001 dolara po stranici Isto važi ako je originalni dokument na papiru pa ga treba prekucati

OTVORENI SISTEMI

Viewer nije kraj priče o otvorenosti sistema Neki sistemi su celi po dizajnu otvoreni I to je dobro Drugi to nisu Recimo de gradimo sistem za naše prilike U smislu pretraživanja teksta, naš jezik sa od engleskog razlikuje po padežima Pitanje padeža se može rešiti pomoću džokera Da samo što to za korisnike nije nimalo praktično

VISUAL BASIC I OPEN TEXT

Obožavam *Visual Basic* Bruka i sramota za nekoga ko je odrastao na asembleru eli sta mogu *Visual Basic* mogu i stere bakice da koriste, toliko je jednostavan Nova verzije *3.0 Professional Edition*, je ne samo jednostavna nego I neverovatno moćne Ako *Visual Basic-u* dodamo parče koda (napisano na C-u) za komunikaciju sa serverom i neko mreže, dobili smo idealnu alatku za razvoj „client-server" aplikacija! *Open Text*, na primer ima nekoliko raznih „client" modula, i svi su napisani na *Visual basic-u* Nje! Nje! Nje! Eto, nisam jedini koji sa pretvono u staru bakicu! V K

Da vidimo Korisnik na „klijentu" otkuca šta želi da pronađa pomoću programa koji se zove „query builder" Taj program zatim šalje „query" serveru Taj „query" može da bude u nekom nerazumljivom kriptovanom obliku e može da bude u razumnom lepo dokumentovanom obliku – kao što je to slučaj sa *Open Text*-om I ne samo to – *Open Text* omogućava i da se na sasvim legalan i elegantan način presretne taj „query" i modifikuje po želji Recimo odsećemo par slove na kraju svake reči i zamenimo ih sa džoker znakom „zvezdica" Gotov posao rešili smo padeže!

Za tako nešto *Excalibur* je otvoren koliko i sef švajcarske banke Doduše *Excalibur* sam rešava pitanje padeže pomoću svoje naurone ke mreže

GRAFIKA I DRUGE SITNICE

Svi sistemi do jednog podržavaju grafiku Recimo, moguće je u tekst stavti nešto poput „vidi sliku" Konsnik dovede miša na to mesto, klikne i pojevi se slike na ekranu To je praktično za velike slike Dovoljno male slike mogu lepo de stanu na stranicu pored teksta

U poslednje vreme je multimedia postala vrlo aktuelna Nije neobično videti nešto poput „vidi film" ili „odslušaj simfoniju"

Takode, većina sisteme dozvoljava i malo naprednije egzibicije Recimo, stavimo u tekst „vidi tabelu" Konsnik klikne na to mesto, program preko mreže otrči do *gateway-a* odatle do velike IBM mašine konsultuje bazu podataka sastavi tabelu, i prikaze je na ekranu

Obično tako nešto zahteva malo programiranje i gotovo svi sistemi su tako dizajnirani da se pomoću njih mogu graditi vrlo složene aplikacije „Infodate" je ne primer prešle sa razvoja programe *Inquire* na razvoj aplikacija za *Open Text Mega Text* i *Folio Views* I tu sa takode može puno zaraditi

KOME OVO TREBA...

.. taj ima para! Tekst-sistemi su obično samo deo celokupnog rešenja Često su originalni dokumenti na papiru Dakle treba ih skenirati e zatim OCR-ovati Skener i OCR sistem koji može da obradi na hiljade stranica dnevno košta oho-ho Ali tu nije kraj priče Na kompjuterima treba skladištiti ne samo tekst nego često i originalnu sliku OCR-ovanog dokumenta To je potrebno zato što OCR tehnologija još uvek nije savršena e i zato što advokati (koji su ovde u Americi bog i batina) uvek vole da vide original Jedne skenirane stranica, kompresovane zauzima oko 50K na disku Pomnožite to sa par stotina hiljada stranica i videćete da je potreban na optički disk neko optički „juke box" Plus još mreža, plus serveri plus povezivanje celog sistema sa nekoliko velikih IBM mašine plus distribuiranje svega toga na više lokacija plus *backup* plus još ovo i ono Poslednji sistem koji sam dizajnirao je koštao sitnih 7 000 000 dolara. Neko ko je speman da potroši tolike pare na kompjutersku opremu verujte mi ima dobar razlog trenutno se jedini pate se mikrofilmovima.

SUDAR SVETOVA

Obećanje objektnog programiranja da će se programi u budućnosti pisati kao što se danas prave elektronski uređaji, jednostavnim slaganjem komponenti, ostvaruje se na najgori mogući način: komponente nove industrije često pate od – nekompatibilnosti (?).

Dejan Jelovčić

Sećate se srednje škole? Ona je bila raz red do vas. Duge noge, plava kosa, uzak struk? Uspeli ste da sredite da vas upoznaju onda ste je pozvali u grad, izveli na večeru i – ništa! Jednostavno se niste slagali. Vi ste voleli da izlazite uveče, njoj su diskoteke bile dosadne. Vi ste hteli da joj objasnite čari košarke nju je više zanimalo da ode u šoping. Njene drugarice su bile dosadne, a vaši drugovi primitivni i naprstonji. Jednostavno, niste se dobro slagali zajedno.

Sećate se srednje škole? Wall, man, imam novosti za tebe: hladan tuš koji si tamo dobio te i sada čeka. Doduše, onda je bila ljubav, sada se radi o poslu. Ovd će manje boleti.

PAKLENI STRINGOVI

Ako ima nešto što tera C++ u minus, onda su to stringovi. Dok je rad sa ostalim tipovima podataka sasvim dobro rešen, C stringovi su prava noćna mora. Ne može se čak ni pisati a=b, već mora da se pozova funkcija za svaku malu sitnicu. Koliko to čini programe nečitljivi jim (doduše i bržim), ne treba ni govoriti. Ako mislite da su razne klase tipa **String** rešenje za ovaj akutni problem, varate se. Probajte samo da definišete strukturu

```
struct primer {

    String a,

};
```

Neće moći! Klasa **String** zahteva da se pozove konstruktor kako bi joj se rekle maksimalna dužina stringa pa bi prema tome, morao de se pravi i konstruktor za celu strukturu. Drugi problem je, recimo, ispisati ovakvu strukturu na disk. Dok su se ranije slogovi upisivali jednostavnim **fwrite** sada je to nemoguće, jer klasa **String** dinamički zauzima memoriju za čuvanje svog sadržaja negde van strukture u kojoj se nalazi, pa bismo sa **fwrite** zapravo, samo upisali pointer na sadržaj stringa dok bi sam string ostao neupisan. Bedak.

Rešenja za **fwrite** postoji iako baš i nije elegantno. Treba samo klasi string omogućiti da čuva svoj sadržaj u nekoj već alociranoj promenljivoj, pomera radi

```
struct primer {

    a_sadrzaj [31],
    b_sadrzaj [21];
    String a,
    String b,
};
```

I kasnije

```
primer::primer (...) . a (a_sadrzaj, 30), b
(b_sadrzaj, 30)
{

}
```

Sada bismo sa **fwrite** upisivali sadržaj sve do promenjive **b_sadrzaj**, a u normalnom radu bismo koristili promenljive **a** i **b**, koje su naravno mnogo komfornije za rad. Može se sa ovom idejom otići još dalje, pa uključiti i nasleđivanje. Sve sadržaje bismo stavili u strukturu **primer_sadrzaj**, a onde bismo iz nje izvel strukturu **primer**, koja bi sadržavala string deklaracije. Time smo obezbedili da sve što nam treba radimo sa strukturom **primer**, a samo upisivanje i čitanje sa diska vrši se strukturom **primer_sadrzaj**. Ne znam koliko je ovo rešenje elegantno (ja bih sigurno bio srećniji sa ugrađenim stringovima), ali važno je da funkcioniše, i to dobro.

KOMPATIBILNOST

Cela ova frka sa stringovima ma dovodi do druge, mnogo važnije teme: kompatibilnosti. Šta ako, na primer, i vaša biblioteka ze rad sa ekranom i vaša biblioteka za rad sa bazama podataka imaju definisane klase **string**. Obe ih trebaju i obe su ih implementirale sa racima malim razlikama. De se odmah razumemo, takva dva biblioteke neće raditi zajedno, jer će se banalno, linker pobuniti zbog duplirania simbola **string**. Cvrc!

Logično rešenje. Pri pravljanju biblioteke ne trebe praviti klasu **String**, već recimo, klasu **dbString**. Sve u našoj biblioteci previmo da ra di sa njom, i onda, na kraju, dodajemo jedno

```
#ifndef DBSTRING
typedef dbString string,
#endif
```

čima klasa **dbString** dobija službeni alias *string* koji je lepši i komforniji za korišćenje, a koji se, naravno, može isključiti jednostavnim stavljanjem

```
#define DBSTRING
```

pre pozivanja hedera. (To će, usput, onemogu ćiti da se heder prekompajlira, ali, hej! svaka se slobode plaća.)

Minus ovakvog rešanja jeste to da ćemo u slučaju da imamo dve ovakve biblioteke, morati da koristimo naziva tipa **dbString** i **scString** u našim programima, umesto jednostavnog **string**. Koliko mi se čini, postoje dve izlaza iz ove situacije:

- Strustrup i ANSI komitet će da standardizuju par osnovnih klasa, kakve su **string**, **bcd**, **vektor** i slično.
- Sve biblioteke ćemo praviti tako da rade samo sa ugrađenim tipovima podataka (**int**, **float**, **char**, ...), a onda će biti ostavljeno krajnjem korisniku na volju, da li će da ih upari sa nekom od **string** klasa.

Drugo rešenje, mada deluje komplikovano, zapravo i nije toliko loše. Sve što treba uraditi kada dokupite novu biblioteku jeste da preklopite sve funkcije koje rade sa tipom **char** da podržavaju i klasu **string**. Nekoliko *inline* poziva bi trebalo da u potpunosti reši problem – ukoliko su programeri spremni da se sa tima zavitlavaju.

Kada bi se krenulo tom stazom, onda bi i prodavci biblioteka odmah uz kod davali i podršku za nekoliko važnijih klasa, tako da bi ceo kod verovatno počeo da liči na UNIX programe

```
#ifdef BIBLIOTEKA1

#else
#ifdef BIBLIOTEKA2

#endif
```

Ružno, zar ne? Ukoliko bi pak Strustrup standardizovao klasu string, spasio bi nas velikih nevolja. Pri tom problem opet ne bi bio do kraja rešen, jer bi se sigurno pojavile i druge klase koje bi pravile nevolje, ali bi sasvim sigurno bio znatno ublažen. Vreme će reći koje će varijanta prevladati.

(Ukoliko sam dobro primetio, Kliparaši na Sezamu trenutno imaju sličan problem. Novi *Clipper 5.2* nema definisan neki simbol koji popularna *NanForum* biblioteka traži, tako da su Klipote primorane na kompromis. Ili novi *Clipper* ili dobra biblioteka. Borbu, čini se, gubi novi *Clipper*.)

KOMPATIBILNOST #2

Što se tiče klasične kompatibilnosti, C++ tu baš i ne stoji dobro. Skoro sam učestvovao u timu koji je trebao da napravi C++ program za više platformi, što bi teoretski trebalo da bude veoma lako ako se ostaje u tekst modu. Teoretski. Ne mogu da nabrojim koliko smo puta udarili glavom o zid pri pokušajima da prenesemo program na neke manje popularne platforme. Jednoj su falili šablonl (templates), druga je loše pravila kopi konstruktore, kod treće je bio bug u biblioteci... spisak je praviše dugačak za ovaj tekst. Pouke su sledeće:

- *Izbegavajte* šablone i izuzetke. Mnogi kompajleri ih još nemaju.
- *Izbegavajte* **iostream** biblioteku. Ovde se možda čak isplati ići u kranost i koristi staru **io.h** biblioteku za sam I/O, jer su čak i neke ANSI implementacije veoma loše i begovite... pogotovu **printf** i **scant** funkcije.
- *Izbegavajte* prenošenje komplikovanih objekata po imenu, radije se oslonite na reference. Dva popularna kompajlera znaju da ubrljaju virtualne tabele u takvim slučajevima. (Gadna buba, usput.)
- *Provucite* sve što napišete kroz *Lint*. To će vas rešiti mnogih glavobolja oko sitnih razlika između platformi.
- Portabilne GUI biblioteke koje obećavaju kula i gradova obično ni na jednoj platformi ne pasuju baš savršeno. Idite pre na interfejs koji je jednostavan i brz nego na *Windows*-like interfejs koji konsnika non-stop muči raznim porukama i dijalozima.

C++ KNJIGE

Pre neki dan sam dobio pismo od momka po imenu Zoran Rilak u kome mi se žali da učenje jezika kakav je C++ bez odgovarajuće literature ide *presporo* Što jes – jes Za one koji imaju mogućnost da nabave knjige iz inostranstva imam tri preporuke

- *The Annotated C++ Reference Manual* Margaret Elis & Bjarne Stroustrup, je knjiga koja služi kao osnova za ANSI standardizaciju jezika I samim tim, neka vrsta C++ biblije Nažalost užasna je za čitanje
- *A C++ Toolkit*, Jonathan Shapiro, je ubedljivo najbolja knjiga za nekoga ko već zna C++ jer vas uvodi u svet objektnog programiranja se stanovišta višestrukog korišćenja Sve fundamentalne strukture (liste nizovi, stabla, hash tabele i slično) u ovoj knjizi su opisane iz objektne perspektive što zasta baca novo svetlo na njihovo korišćenje u svakodnevnim programima
- *C++ Primer*, Staley Lipman, je definitivno najbolja knjiga za nekoga ko želi da nauči C++ Visoko preporučeno!

Ukoliko ste pak, osuđeni na domaću literaturu, onda je jedina alternativa Strustrupova knjiga *C++* u izdanju Mikro Knjige Ova knjige sa bavi starom verzijom jezika i loše je napisana, no u svoje je doba bila ono što je *The Annotated C++ Reference Manual* danas – biblija U svakom slučaju, knjiga će poslužiti sasvim dobro nekome ko je spreman da se rve sa Strustupovim neujednačenim stilom

Ako ste tanki s parama možde bi trebali da se odlučite za neki tekst u elektronskoj formi Oni, naravno, ne mogu ni da se približe dobroj knjizi (de mogu, bili bi objavljeni kao knjiga) ali rade posao sasvim prihvatljivo Za sada mi padaju *C++ Tutoriel* kao i *C++ FAQ* deloteka kao odlični izvori informacija o ovom jeziku a čini mi se de sem ne Internetu video još dosta toga. Ukoliko imate pristup nekom VAX-u eto zabave za vas!

AKO VAS JE BRIGA

Moj komšija ima na svojim kolima novi ukras rupu od metka Pošto dva puta nije pustio nekog lipa de ga pretekne da ne bi došlo do sudara, ovaj je izvadio pišolj i uz kola pucao na njega! Sa druge strane voleo bih i ja de imam na kolima rupu od metka jer za sada nemam ni kola – ukradena su Dva moja prijatelja su u poslednjih godinu dana opljačkana na ulici, e pištolj je postao uobičajeni modni element (Čeka se trenutak kada će oružje početi da se prodaje u raznim bojama i dezenima tako da ga možete nositi uz razna odela)

Trenutno, panzionen jure golubove na Kalemegdanu za večeru, prosečna plata je ispod deset maraka seljaci su pređeni za pšenicu i deca umiru po bolnicama Pitanje je, da li se uopšte isplati učiti C++? Možda je bolje da uložite svoje pare u bolju bravu i metalne rešetka na prozorima?

Bože, kako se nadam da nije došlo takvo vreme!

C MOZGALICE

IMATE LI OKO SOKOLOVO?

Ako ste C programer, onda znate koliko se skrivenih buba može kriti u svakom vašem programu. Neki čak idu dotle da kažu da se kvalitet C programera prepoznaje po tome koliko brzo nalaze greške u kodu. „Protrčali" smo kroz uputstvo za *Lint* i pronašli u njemu primere parčiće koda koji ne daju očekivane rezultate. Oprobajte svoje znanje!

Listing 1

```
int main ()

{
    char ch1 = '\xOF';
    char ch2 = '\xFF';
    unsigned u = (ch1 << 8) | ch2
    n = u & ch2;
    printf ("%x\n", u);
    return 0;
}
```

Ovej program ne štampa „F00". Zašto?

Listing 2

```
#include <stdio.h>

void patuljci (int ko[i]

{
    switch (ko[i]) {
        case 1: printf ("Ljutko"),
        case 2: printf ("Kijavko"),
        case 3: printf ("Trapavko"),
    }
}
```

Osim Snežane i još nekoliko patuljaka, ovom programu fali još nešto Šta?

Listing 1

```
#include <stdio.h>

main ()

{
    char ch = 0xFF;

    if ((unsigned)ch == 0xFF)
        printf ("Jednake"),
}
```

Šta štampa ovaj program?

Listing 4

```
#include <stdio.h>

main ()

{
    unsigned row = 25, col = 80,

    /* Neki kod... */
    if (row + col < 1)
        printf ("greška u podesima\n"),
}
```

Nešto je veoma neloglčno u ovom programu Znate li šta?

Listing 5

```
int daljenje (double x)

{
    int n = 0,

    while (x > 1/10) {
        x /= 2,
        n++,
    }
    return n,
}

int main ()

{
    printf ("%d\n", daljenje (1)),
    return 0
}
```

Neverovatno, ali ovej program štampa broj veći od hiljadu, umesto de štampa 16 Gde je greška?

Listing 1

```
int a [2048],

void init ()

{
    int i,

    for (i = 0, i < 2048, i++),
        a [i] = i,
}
```

Umesto da se inicijalizuje niz *a*, on ostaje popunjen nulama Zašto?

Listing 7

```
unsigned make_word (unsigned hi, unsigned low)

{
    return hi << 8 + low,
}
```

Ovoj funkciji baš i ne uspeva da upari više i niži bajt. Zašto?

Listing 8

```
#include <stdio.h>

void g (int n)

{
    printf ("%d\n", n),
}

int main ()

{
    int n = 371,
    g (n / 10),
    return 0,
}
```

Umesto da štampa 37, ove funkcije vraća samo 3. Imate li oko sokolovo?

Dejan Jelovac

OBJEKTI ZA NESTRPLJIVE

Ako vam je dosadilo čekanje na *Clipper* sa „pravim" objektima, onda pročitajte sledeći tekst: u njamu će biti opisano kako se od običnog *Clipper*-a može dobiti pravi objektni jezik.

Nenad Bafundin

Objekti koji postoje u *Clipper*-u su prilično čudno rešeni postoji samo nekoliko ugrađenih klasa i programer može kreirati i koristiti samo objekte koji pripadaju tim klasama. Kreiranje novih klasa nije moguće. Neobičnost se sastoji u tome što *Clipper* sadrži interne funkcije koje omogućavaju kompletan rad sa objektima, kreiranje novih klasa, slanje poruka itd – sve je na svom mestu. Do mogućnosti pravog objektnog programiranja ostao je samo još jedan mali korak. Međutim, auton *Clipper*-a su iz nekog razloga smatrali da taj prelaz ne treba još uvek dopustiti korisniku. Naravno, to će se sasvim sigurno dogoditi u narednoj verziji *Clipper*-a, ali dotle treba čekati.

Sa druge strane, objekti su zaista nasušna potreba u tipičnim *Clipper* programima, to su poslovne aplikacije koje je potrebno relativno često menjati. U takvim programima se često javljaju delovi koji se sasvim malo razlikuju od nekog već postojećeg koda u istom programu. Tipičan primer su razni tabelarni prikazi, učitavanje i izmena slogova iste strukture, itd. Proširivanje liste argumenata postojećih procedure donekle rešava problem. Međutim, takvi jednostavni indikatori stanja ne daju uvek dobre rezultate i programer obično tada koristi *cut & paste* funkcije editore da bi realizovao sličan deo koda. Tako nastali delovi su veoma česti u *Clipper* programima. Bez obzira što je ovakvo rešenje verovatno najjednostavnije i najbrže u prvi mah, kasnije ono zadaje najviše glavobolje. To se vidi prilikom prve modifikacije iskopiranog dela programa: izmene se moraju paralelno uraditi na svim iskopiranim delovima, a to može biti ogromen i strahovito zamoran posao. Uz to, pri mehaničkom ponavljanju operacija lako dolazi do raznih grešaka, čime sve to postaje veoma loše po program. Objekti u ovakvim slučajevima mogu biti spasonosno rešenje: objektna organizacija programe obično dovodi do smanjenog ponavljanje istih delove koda, tj. jeden kod se mnogo jednostavnije koristi višestruko.

Objekti takođe omogućuju lakšu nadgradnju preko nasleđivanja i predstavljaju odličnu alternativu za standardno modularno programiranje. Pareleino održavanje više sličnih verzija programe je takođe jedna od objekti mogu pokazati svu svoju fleksibilnost i moć.

PRAVI I PSEUDO-OBJEKTI

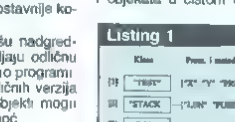
Ali, svaj sve to gubi na značaju pošto *Clipper* ne omogućava programeru definisanje novih klasa. Preciznije, sve potrebne funkcije postoje, ali nisu nigde spomenute. Na primer, u dokumentaciji **Extend** sistema mudro se zeobilazi da postoji tip **object** može se raditi sa nizovima, stringovima i detumima, ali ne i sa objektima!

To što se dokumentacija ne može nabaviti ne znači previše: dosta firmi koje proizvode dodatke za *Clipper* saznalo je kako se te funkcije koriste i to je bilo dovoljno. Vrlo brzo su se pojavile biblioteke za *Clipper* koje su omogućava le korisniku da kreira nove klase i objekte. Najpoznatije takve biblioteke su *o.Clip*, *Class(y)*, *SuperClass*, itd. Nedavno smo se u „Računanama" upoznali sa jednom od njih (*o.Clip*). Većina tih biblioteka koristi nedokumentovane interne *Clipper*-ove funkcije za kreiranje novih klasa. Objekti koji se kreiraju na osnovu tih klasa se ni po čemu ne razlikuju od „fabrički" ugrađenih objekata. Performanse su uglavnom zadovoljavajuće. Osnovni problem kod korišćenje ovih biblioteka je nešto drugo: prelazak na novu verziju *Clipper*-a. Naime, niko ne garantuje da će *Clipper* i u narednoj verziji imati iste interne funkcije, pa je za očekivati da biblioteka koja radi sa tekućom verzijom ne radi sa narednom verzijom *Clipper*-a. Da stvar bude još gora, auton ovih biblioteka namerno vezuju kod za tekuću verziju *Clipper*-a, kako bi naterali korisnika na novu kupovinu. Cena ovih biblioteka je srazmerno niska, čak i za naše uslove, ali problem je nabavka. Još gore je što ovakve biblioteke proizvode male firme koje mogu lako nestati, a onda možeta biti ostavljeni na cedilu sa verzijom koja ne radi.

Sledeći problem je prelaz ne verziju *Clipper*-a koje će omogućavati kreiranje objekata, tada valja prepraviti stare programe, a ko zna koliko će to biti komplikovana operacija. Zbog svega toga, ostajanje na neku ovakvu biblioteku može biti prilično bolno. U suštini verovatno će pojava *Clipper*-a 6.0 ugasiti ovakve proizvode, ali nište nije sigurno, auton *Clipper*-a su uvek namerno ostavljali prostor za razne dodatke!

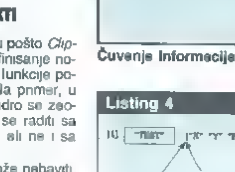
Međutim, za pravi objektni *Clipper* se sigurno valja pripremati: organizacija programa će morati da pretrpi dosta izmena da bi se preslo na objekte. Verovatno iz prethodno pobrojanih razloge mnogo programera koristi alternativu previh objekata, tzv. **pseudo-objekte**. To je naziv za čitavu grupu veoma raznolikih rešenja, ali u principu se odnosi na realizaciju klasa i objekata u čistom *Clipper*-u, bez korišćenje

Listing 1



Čuvanje informacije o klasama

Listing 4



Primer dve objekta klasa *TEST*

interpih funkcija. Ovakva rešenja imaju dosta mana, ali i poneđ toga su prilično popularna, između ostalog i zbog sigurne prohodnosti u narednu verziju *Clipper*-a.

U ŠIROKOM KRUGU

Znači, napravićemo proširenje *Clipper*-a koje omogućava kreiranje novih klasa i objekata, zaobilazeći postojeći *Clipper*-ov sistem. Analizom postojećih rešenje može se videti da sva konste prilično nestandardizovane ideje. Ali u osnovi sva konste isti princip, pošto nije moguće kreirati prave objekte potrebno je kreirati objekte koristeći raspoložive tipove podataka. Naravno, kao najlogičniji kandidat se nameće niz: objekt i niz su slični, jer i jedan i drugi sadrže više podataka raznih tipova. Osnovna razlika je u tome što objekti, za razliku od nizova, „znaju" i **nazive** svakog svog elementa. Tako će primera radi **TBrowse** objekat **TB** savršeno „razumeti" konstrukciju **TB:colPos**, jer postoji evidencija o tome koja interna promenljiva ima naziv **colPos**. **TBrowse** objekat se može posmatrati i kao niz, s tim što informacije o nazivima elemenata u nizu ne postoji niz „zna" samo redni broj elemenata niza. Znači, objekti se ne mogu jednostavno preslikati u nizove.

Ostavićemo objekte za trenutak i vratićemo se na početak problema: kreiranje **klase**. Klasa je definicija grupe objekata: ona sadrži informacije o internim promenljivim i funkcijama svakog objekta iz ta klase. Na primer, svaki objekat *Clipper*-ova ugrađene klase **TBrowse** ima iste interne promenljive i funkcije. Kasnije se mogu menjati vrednosti tih promenljivih, ali struktura uvek ostaje ista.

Klase ćemo prirodno predstaviti korišćenjem jednog niza u kome svaki element opisuje jednu klasu. Svaki član niza će sadržati tri elementa: naziv klase, niz naziva internih promenljivih i funkcija (metoda) i niz koji čuva inicijalne vrednosti. Na slici 1 je prikazana struktura takvog niza **aClass**. Kada se kreira nova klase, svi podaci o njoj se upisuju u niz **aClass**. Na primer, ako definišemo sledeću klasu:

```
CLASS Test
  VAR x
  VAR y  = 11
  METHOD Print
END CLASS
```

onde će odgovarajuća trojka biti

```
{
  "TEST",
  {"X", "Y", "PRINT"},
  {NIL, 11, NIL}
}
```

Nešto kasnije ćemo detaljno analizirati prethodnu definiciju. Za sada je dovoljno da znamo da se tako definiše nova klasa. U definiciji stoji da se klasa naziva **Test**, da ima dva interne promenljive **X** i **Y** i jednu internu funkciju **Print**. Takođe je zadato da promenljiva **Y** ima inicijalnu vrednost 11. Znači, svaka nova klasa se upisuje u niz **aClass** kao jedan niz koji se sastoji od tri elemnta: jednog stringa i dva niza. Ovom rešenju se može ponešto i dodati. Na

Listing 2

```
// Klase
STATIC aClass := {}

*****************************************************************
****
**** DATUM: 12.07.1993
****
**** NAZIV: __CreateClass (cClassName)
****
**** OPIS:  Kreiranje nove klase
****
****
*****************************************************************

PROC __CreateClass (cClassName)

   // Ispitaj samo ako vec ne postoji klasa sa datim imenom
   IF AScan (aClass, {|e| e[1] == Upper(cClassName) } ) == 0
      AAdd (aClass, {Upper(cClassName), {}, {}})
   ELSE
      __Error ("Klasa " + cClassName + " vec postoji !")
   END IF

RETURN



*****************************************************************
****
**** DATUM: 12.07.1993
****
**** NAZIV: __AddVar (cClassName, cVar, xInit)
****
**** OPIS:  Dodaje promenljivu u definiciju klase
****
****
*****************************************************************

PROC __AddVar (cClassName, cVar, xInit)

   LOCAL nPos

   // Nema sta ako klasa postoji
   IF (nPos := AScan (aClass, {|e| e[1] == Upper(cClassName) } )) <> 0
      AAdd (aClass[nPos,2], Upper(cVar))
      AAdd (aClass[nPos,3], xInit)
   ELSE
      __Error ("Klasa " + cClassName + " ne postoji !")
   END IF

RETURN
```

Funkcije za kreiranje nove klase

Listing 3

```
*****************************************************************
****
**** DATUM: 12.07.1993
****
**** NAZIV:  o := __nCreateObject (cClassName)
****
**** OPIS:  Kreira objekt iz klase cClassName
****
* **
*****************************************************************

FUNC __CreateObject (cClassName)

   LOCAL nPos, aAss

   // Da li klasa postoji?
   IF (nPos := AScan (aClass, {|e| e[1] == Upper(cClassName) } )) <> 0
      aRet := { aClass[nPos,2], AClone(aClass[nPos,3]) }
   ELSE
      __Error ("Klasa " + cClassName + " ne postoji !")
   END IF

RETURN nPos
```

primer nigde ne postoji informacije o tome da je **Print** funkcija, a ne interna promenljiva klase **Test** – bilo bi zgodno dodati informaciju o tome. Ipak, u radu se retko greši tako što se zamene interne funkcija i promenljive, teško da ova osobina i nije previsoko ograničenje, mada bi rešenje koje pravi razliku između promenljivih i funkcija bilo kvalitetnije jednostavno. Što se tiče internih promenljivih, bilo bi veoma korisno dodavanje informacije o vidljivosti u odnosu na ostale klase.

Kada se definiše klasa, treba navesti koje su njene interne promenljive i funkcije. Za prethodni primer, to su promenljive **X**, **Y** i funkcija **Print**. Na slici 2. su prikazane funkcije za kreiranje nove klase i dodavanje internih promenljivih.

Funkcije **__CreateClass** upisuje novu klasu u postojeći spisak, tako što prvo proveri da li postoji klasa sa zadatim imenom i u tom slučaju javlja grešku. Ako je sve u redu (klasa sa istim imenom ne postoji), u niz **aClass** se upisuje naziv nove klase i dva prazna niza. Interne promenljive i funkcije se dodaju pomoću funkcije **__AddVar(cClass, cVar, xInit)**, koje

Kreiranje objekata

Funkcije za pristup pseudo-objektima

Listing 5

```
*****************************************************************
****
**** DATUM: 12.07.1993
****
**** NAZIV:  v := __VarAssign (oObject, cVar, xVal)
****
**** OPIS:   oObject cVar = xVal
****
****
*****************************************************************

FUNC __VarAssign (oObject, cVar, xVal)

   LOCAL nPos

   // DA LI postoji promenljiva?
   IF (nPos := AScan (oObject[2], Upper(cVar))) <> 0
      oObject[2,nPos] := xVal
   ELSE
      __Error ("No postoji promenljiva " + cVar)
   END IF

RETURN xVal



*****************************************************************
****
**** DATUM: 12.07.1993
****
**** NAZIV:  v := __VarGet (oObject, cVar)
****
**** OPIS:   v := oObject:cVar
****
****
*****************************************************************

FUNC __VarGet (oObject, cVar)

   LOCAL nPos, Ret

   // Da li postoji promenljiva?
   IF (nPos := AScan (oObject[1], Upper(cVar))) <> 0
      Ret := oObject[2,nPos]
   ELSE
      __Error ("NA postoji promenljiva " + cVar)
   END IF

RETURN Ret



*****************************************************************
****
**** DATUM: 12.07.1993
****
**** NAZIV:  o := __Self()
****
**** OPIS:   Vraca trenutni aktivni objekt
****
****
*****************************************************************

FUNC __Self

   LOCAL nPos

   IF Len(aActObject) <> 0
      oRet := aActObject[Len(aActObject)]
   ELSE
      oRet := NIL
   END IF

RETURN oRet



*****************************************************************
****
**** DATUM: 12.07.1993
****
**** NAZIV:  __Send (oObject, cMethod, oObject)
****
**** OPIS:   Izvrsava metod
****
****
*****************************************************************

FUNC __Send (oObject, cMethod, oMethod)

   LOCAL nPos, oRet

   // Upisi aktivni objekt na stek
   AAdd (aActObject, oObject)

   // DA LI postoji metod ?
   IF (nPos := AScan (oObject[1], Upper(cMethod))) <> 0
      oPos := Eval (oMethod)
   ELSE
      __Error ("Method " + cMethod + " ne postoji ! ")
   END IF

   // Skini aktivni objekt sa steka
   ASize (aActObject, Len(aActObject) -1)

RETURN xRet



*****************************************************************
****
**** DATUM: 12.07.1993
****
**** NAZIV:  __Error (cTxt)
**
**** OPIS:   Varijaa o greski
****
****
*****************************************************************

FUNC __Error (cTxt)
   Alert (cTxt)
   QUIT
RETURN
```

klasi sa nazivom **cClass** dodaje internu promenljivu **cVar** koja ima inicijalnu vrednost **xInit**. Na primer **__AddVar("TEST", "Y", 11)** dodaje klasi **Test** internu promenljivu **Y** se inicijalnom vrednošću 11. Pošto se isti zapis koristi i za interne funkcije funkcija **__AddVar** se može koristiti i za dodavanje imena internih funkcija. Na primer

__AddVar ("TEST", "PRINT", NIL)

definiše de klase **Test** ima internu funkciju **Print**. U strožijoj realizaciji funkcija **Print** bi morala biti „zatvorena" u okviru klase. Ij. ne bi postojala mogućnost da se ova funkcija poziva van objekata iz klase **Test**. To u ovom rešenju nije urađeno, ali bi se pomoću pretprocesora moglo automatski dodavati neki pretiks nazivu funkcije koji bi lako funkciju „sakno" od ostatka programa.

OBJEKTI

Sledeći korak je definisanje objekata. Na žalost u čistom *Clipper*-u nikako nije moguće napraviti objekte koje će i *Clipper* prepoznavati kao objekte. Čak ni u asemblera to nije moguće bez korišćenja nedokumentovanih funkcija. Zato nam jedino preostaje da za objekte nademo neki zgodan alternativni način prikazivanja. Tu se kao prvo rešenje nameću nizovi po strukturi najsličniji objektima. Međutim, prosto preslikavanje objekata u nizove nije moguće, objekat mora imati i informaciju o nazivima svojih internih promenljivih i funkcija. Zato se kao jedno rešenje može uzeti niz sastavljen od dve niza: niza naziva promenljivih i niza odgovarajućih vrednosti

```
{
  {"X", "Y ', "PRINT"},
  {NIL, 11, NIL}
}
```

Znači, kada se kreira jedan objekat klase **Test**, kao rezultat će se dobiti opisani niz. Na primer

```
o := Test()
```

kreira novi pseudo-objekat **O** koji ima baš prethodno prikazanu strukturu. Tehnički gledano ovakav objekat se kreira veoma lako: prostim kopiranjem odgovarajućih nizova iz definicije klase. Međutim, tu postoji jedan problem: drugi niz čuva vrednosti i svaki objekat mora imati svoju nezavisnu kopiju. Na primer, ako posle prethodne naredbe napišemo **p := Test()** i promenljiva **P** će da dobiti kao vrednost identičnu strukturu. Međutim, ako se napiše nešto poput **p:y := 22**, interna promenljiva pseudo-objekta **Y** dobiće vrednost **22**, ali će zato vrednost u pseudo-objektu **X** ostati neizmenjena. Prva komponenta – niz koji čuva nazive je potpuno drugi slučaj, taj niz ostaje nelzmenjen za sve objekte jedne klase, bez obzira na izmene vrednosti objekata. Može imalo bi smisla razmisliti o dinamičkoj izmeni strukture objekata, ali to već izlazi iz naše teme.

Znači, svaki pseudo-objekat ima potpuno istu kopiju niza se imenima internih promenljivih i funkcija. To u slučaju klasa/objekata se više promenljivih može biti značajan memorijski prostor. Zato bi bilo dobro kada bi svi objekti jedne klase mogli da koriste jednu kopiju niza. Naravno najprodnije je da ta kopija bude upravo druga komponenta definicije klase u nizu **aClase**. Na slici 3. je prikazana funkcija **__CreateObject** koja za zadati naziv klase kreira pseudo-objekt. Prvi elementi tog pseudo-objekta je **pokazivač** na niz koji sadrži nazive promenljivih i funkcija dok je drugi posebna kopija niza vrednosti (za kopiranje niza se koristi funkcije **AClone**). Na slici 4. je prikazan primer strukture definicije klase **Test** i definisana dva objekta te klase.

KAKO IM PRIĆI?

Kao što je već rečeno opisani sistem ne kreira prave objekte već najobičnije nizove tako da uobičajena sintaksa za pristup elementima pseudo-objekta ne dolazi u obzir. Zalsta šteta jer originalni *Clipper*-ov **send** operator je veoma leško zamenili nečim tako jednostavnim i elegantnim. Na primer izmena interne promenljive objekta **TB** iz klase **TBrowse** jednostavno se radi ovako

TB:colPos := 20

Simbol „:" se naziva *send* operator i on se univerzalno koristi za pristupanje svim elementima objekata. Na primer interne funkcija za regenerisanje ekrana se poziva se

TB:refreshAll()

Ovakvu jednostavnost je nemoguće postići sa pseudo-objektima i to je izuzetan nedostatak ovih sistema koji koriste pseudo-objekte. Pretprocesor ima kako ćemo malo kasnije videti veoma značajnu ulogu u „skrivanju" tehničkih detalja oko realizacije pseudo-objekata ali je u slučaju send operatora nemoćan. Izraz **TB:colPos** je regularan *Clipper* izraz i njegova analize pretprocesorom kao izdvojene neredbe je nemoguća. Nsme, ako pokušate da definišete nešto poput

#command <o>:<m> => Send (<o>, <m>)

i tako slanje poruke preusmerite na vašu funkciju, necete uspeti: jednostavno, neredba tipa **TB:refreshAll()** neće biti obrađena!

Listing 6

```
//
// CLASS Stack
//
#command CLASS <name>   =>
         PUBLIC <name>
            STATIC _iDefined := .F.
            LOCAL _oName := <name>():New()
            IF ! _iDefined
               _iDefined := .T.
               _CreateClass (_oName)
            ENDIF
            RETURN _oName:CreateObject()
//
// VAR x, y := 100, z
//
#command VAR <v> [ := <ini> ] [, <v2> ] => ;
         _AddVar (_oName, <"v">, NIL) ;
         [; VAR <v2> ]
//
#command VAR <v> := <vp> [, <v2> ] => ;
         _AddVar (_oName, <"v">, <vp>) ;
         [; VAR <v2> ]
//
// METHOD Push( x )
//
#command METHOD <m> [, <m2> ] => ;
         _AddVar (_oName, <"m">, NIL) [; METHOD <m2>]
//
// END CLASS
//
#command END CLASS      => ;
         END IF ;
         RETURN _oCreateObject(_oName)
//
// Assign [:=assign] "Para"
//
#translate Assign [<:=:>] <p> => ;
         _VarAssign( <"m">, <p> )
//
// ? Retrieve [:=arge]
//
#translate Retrieve [<:=:>] => _VarGet( <"m"> )
//
// ? Send [:=print](1,2,3)
//
#translate Send [<:=:>] => Send( <o>, <"m">{} )
#translate Send [<:=:>](<list,...>) => ;
         _ASend( <o>, <"m">, {|| <list>} )
//
// Assign ):= Low )
//
#translate <o>:<"m"> := <Self>:<"m">
```

Heder za nove komende za rad sa objektima

Jedini način da se ovo donekle ublaži je sistematizacija postojećih operacija nad objektima. Bez obzira šio se uvek koristi jedan operator **send** operator se koristi za tri operacije

- **Čitanje vrednosti interne promenljive**. Na primer, ako želite da odštampate internu promenljivu **colPos** objekta **TB** napisaćete **? TB:colPos**. Za ovu operaciju je zadužena funkcije **__VarGet** prikazana na slici 5. Ova funkcija za zadati pseudo-objekt i naziv njegove interne promenljive vraca odgovarajuću vrednost. Na primer **__VarGet(o, "Y")** vraliće vrednost 11
- **Dodeljivenje vrednosti internoj promenljivoj**. Na primer, ako želite da promenljivoj **colPos** objekta **TB** dodelite vrednost 20 napisaćete **TB:colPos := 20**. Za ovu operaciju zaduzena je funkcija **__VerAssign** (slika 5). Ova funkcija za zadati pseudo-objekat, naziv interne promenljive i vrednost dodaljuje vrednost internoj promenljivoj. Na primer **__VarAssign(o, 'Y", 33)**
- **Aktiviranje internih funkcija**. Interne funkcije objekta se uobičajeno nazivaju **metode** a njihovo aktiviranje se naziva **slanje poruke**. Na primer, **TB:refreshAll()** je izraz koji označava de se objektu **TB** šalje poruka **refreshAll** koja mu nalaže da regenerise prikaz na ekranu. Za ovu operaciju je zadužena funkcija **__Send** sa slike 5. Ona prvo proverava de li se u spisku internih funkcija objekte nalazi i traženi metod e zatim ga aktivira. Metod se zadeje kao kodni blok. Na primer, ako treba poslati poruku **Print()** treći argument funkcije **__Send** biće { || Print() }

Samo aktiviranje metoda nije problem, samo treba upotrebiti standardnu funkciju **Eval**. Međutim, jevlja se nešto drugo: metod je najobičnija funkcija koja moze raditi proizvoljnu operaciju. Pn tom se mogu koristiti svi raspoloživi *Clipper* operaton. Izrazi funkcije itd. Jedina razlika u odnosu na standardnu funkciju je to što metod pripada odredenom objektu pa mora imati mogućnost pristupa internim promenljivim tog objekta. Kao moguce rešenje za to, može se uvesti da svaki metod mora imati kao prvi argument objekat kojem pripada: tako bi se problem rešio, ali bi se nepovratno izgubile elegancija definisanje metoda.

Klasični objektni jezici pozniju termin **Self** koji označava objekat kojem pripada metod. Na primer, ako se pri definisanju metoda **Print** iz objekta **O** referišete na objekat **Self** to će biti upravo objekat **O**. Ova definicija je važna jedino u trenutku izvršavanja metode. Funkcije **__Send** zato, pre nego sto aktivira metodu, određuje aktivan objekat (u stvari objekat kojem pripade metoda koja se aktivira). Pošto se može desiti da se u okviru jedne metode pozove druga metoda nekog drugog objekta, aktivni objekti se stavljeju na stek, pri čemu je uvek na vrhu trenutno aktivan objekat. Kada se metoda izvrši, objekat se skida se steka. Funkcija **__Self** vraća objekat se vrha steka tj. trenutno aktivni objekat.

AMBALAŽA...

Pobrojane funkcije omogućavaju osnovne operacije se objektima i klasama: definisanje novih klase, kreirenje objekata, rad sa internim promenljivim i metodama objekta i sl. Međutim, teško da bi se iko odlučio na korišćenje ovako nezgrapnih funkcija u tu svrhu. Dakle, iskonstićemo *Clipper*-ov pretprocesor i upakovaćemo opisane funkcije u lepšu ambalažu. Na slici 6. je prikazan heder datoteka koji sadrži sve potrebne definicije. Prvo cemo rešiti

kreiranje novih klasa. Umesto pozivanja funkcija __CreateClass i __AddVar konstelemo mnogo prirodnije i lepša konstrukcija CLASS END CLASS i sl. Na primer, pretprocesor će pomenuti definiciju klase Taat prevesti u

```
FUNC Test
  STATIC __lDefined := .F.
  LOCAL __cName := "Test"
  IF ! __lDefined
     __lDefined := .T.
     __CreateClass (__cName)
     __AddVar (__cName, "x", NIL)
     __AddVar (__cName, "y", 11)
     __AddVar (__cName, "Print", NIL)
  end
RETURN __CreateObject (__cName)
```

Pretprocesor definiciju klase pretvara u definiciju istoimene funkcije koja služi i za definisanje klasa i za kreiranje objekata te klase. Sada će prvi poziv o := Test() pozvati funkcija __CreateClass i __AddVar i tako upisati kompletnu definiciju klase u odgovarajući niz. Na kraju će se pomoću funkcije __CreateObject kreirati novi objekt te klase i to će biti razultat funkcije. Kada sa ponovo pozova funkcija Test() (ona se naziva **konstruktor** jer kreira nove elemanta klase – objekta) neće sa ponovo pozivati funkcije za definisanja klase statička promanljiva __lDefined je u prvom prolazu dobila vrednost TRUE tako da će svaki sledeći poziv samo kreirati novi elemenl klase Test.

Funkcija za čitanja i dodeljivanje vrednosti lakode dobijaju svoja zamena. Za čitanje interne vrednosti objekta zadužena je funkcija **Retriave**. Na primer **Retrieve(o x)** je ekvivalentno vrednosti **o.x** – vrednost promenljiva **X** objekta **O**. Primalite da se u okviru neke delini sene komande može redefinisati send operator "." za razliku od semostalne naredbe.

Za dodelu vrednosli se koristi funkcija **Aasign**. Na primer, **Aaslgn (o x, 'ABC")** odgovara zapisu **o:x := 'ABC'** – interna promanljiva **X** objekta **O** dobija vrednost **"ABC'**

Aktiviranje interna funkcija (metode) nekog objekta obavlja se funkcijom **Send**. Na primer, može se pisati

```
Send (o:print())
```

Ovim se objektu **O** „naređuje" da aktivira internu funkciju **print**. Ovako se mogu slati i poruka sa argumantima. Na primer

```
Send (o:print(1,"ABC"))
```

Funkcija **Send** vraća razultat funkcija **print()**, pa se može upotrabljavali i u okviru proizvoljnih izraza, a na samo kao posabna naredba. Interesantna je realizacija preslikavanja u funkciju **__Send** pomoću pretprocesora. Taj posao obavlja sledeća direktiva

```
#translate Send (<o>:*<m>([<list, ,>])) => ;
   __Send(<o>, <"m">, {|| <m>(<list>)} )
```

Značl prethodna naredba se pravodi u

```
__Send(o, "print", {|| print(1,"ABC")} )
```

Ovde se lapo vidi elegancija pretprocesora prilikom analize izraza iz **print(1,"ABC")** smo dobili informacije i o nazivu funkcija i o argumantima bez ikakvih posebnih analiza.

Još je preostalo referansanja na „maličnl" objekat iz meloda. Za to je zadužena konstrukcija „:". Na primar, metoda **print** za klasu **Test** se može definisati na sledeći način

```
PROC Print
  ? "Interno X =", Retrieve(::x)
  ? "Interno Y =", Retrieve(::y)
  Aaslgn (:. x, Retrieve(::x) + 1)
RETURN
```

Znači svako aktiviranje metoda **print** štampa vrednost promanljivih **X** i **Y** objekta a zatim uvećava vrednost **X** za jedan

NASLEĐIVANJE

Opisani sistem se može sasvim lapo upotrebljavati u praksi, ali pri toma treba voditi računa o nakoliko važnih stvan. Prvo, traba imati u vidu da je pristupanja alamenjima objekata dosta sporo, jar sa uvak pri čitanju/upisivanju vrednosti prvo izvršava funkcija **AScan** koja služi za proveru da li referisana promenljiva uopšte postoji. To znači da intema promenljiva pseudo-objekata ne treba konstiti u ciklusima niti uopšte tamo gde je potraban veoma brz pristup vrednostima. Isto vazi i za aktiviranja metoda. uvak će se izvršiti nekoliko poziva više pa će ukupno vreme izvršavanja biti produženo. Performanse sa značajno mogu poboljšati optimizacijom funkcija za pristup objektima – ovde je zbog jednostavnosti korišćen čistji i sporiji kod.

Drugi problam je odnos prama „fabrički" ugrađenim klasama **Error Get** i **TBrowse**. Direktna veza na postoji i jedino rešenja je da sa funkcija za pristup objektima prerade. Iako da prvo provari da li se operacija odnosi na pseudo-objekat ili na pravi objekat i u zavisnosti od toga da se izvrši odgovarajuća operacije. Naravno ovim bi se malo uspono pristup ograđenim objektima.

Treći problem je veoma vazna mogućnost pravih objektnih sistema, a to je **nasleđivanja**. To je mogućnost da pri definisanju nove klase iskonsitita definicije već postojećih. što ja dragoceno pri širenju postojeće hijerarhije objekata. Na primer, definisali ste neki objekat koji obavlja neku funkciju u sistemu. Ako vam zatreba objekat koji isto radi kao i definisani, sa nekom malom razlikom (problam izmene postojećeg kodal) može se definisati nova klasa kao **naslednik** neka od postojećih klasa. Objekti iz nova klase „znaju" sve što i objekti-roditelji, ali ih proizvoljno možete nadograđivati, bez izmene osnovne klase.

Najjednostavnije nasleđivanja (koja se česlo sreća) veoma se lako realizuje. Jednostavno se pri definiciji klasa dodaju sve interne promenljiva i melode klase koja se nasleđuje. Međutim, šta sa promenljivim i metodama istog imena? Radi se o lome da se pri nasleđivanju klasa potpuno sačuva funkcionalnosl objekata iz roditeljska klase. I ovo nije lako komplikovano realizovati, ali bi znatno iskomplikovalo postojeće rešenje koje. Ipak, treba da posluzi samo kao primer. Sama realizacija nasleđivanja (pa čak i višestrukog nasleđivanja) nije previše kompliKovana samo treba uvesli dodatne pokazivace na definicije „roditaljskih" objakata.

Opisani sistem je potpuno funkcionalan i možete ga slobodno konstili u vašim programima s tim da obratite pažnju na napomene o performansama. Pošlo su sve procedura napr. sane u čistom *Clipper*-u, sva je pod vašom potpunom kontrolom, ako vam zatreba neko poboljšanje. Iako cela ga ugradili u sistem. Ko zna – možda će opisani sistem biti dovoljno rešenje da bi prešli makar delimično na **OOP**, možda će učinit da poboljšate strukturu programa, a možda će vam jednostavno skratiti čekanja do pojave prava objaktna varzije *Clipper*-a!

STA IMA NOVO

SOFTVER

Stono izdavaštvo

Majkrosoftov bestseler: Publisher 2.0

12. jula zvanično je predstavljen *Microsoft Publisher 2.0* za *Windows*. Nova verzija paketa za stono izdavaštvo sa preko 25 novih mogucnosti. Objedinjujući sofisticirane funkcije i jednostavnost korišćenja, *Publisher* omogućava različitim vrstama korisnika da kreiraju publikacije visoko-profesionalnog kvaliteta – od jasne letke, prospekte, brošura, pozivnice, čestitke, formulare, kalendare – i to samo pomoću nekoliko komandi.

Jedna od novih opcija je *QueCards on-line* savetnik koji se otvara jednim klikom miša i pruža korisniku svu potrebnu pomoć u dizajniranju, bez potrebe da napušta publikaciju u kojoj radi. Tu je i *Page Wizard* dizajnerski pomoćnik, obilk interaktivnih šablona (*templates*), koji korisniku poseduje niz pitanja u vezi sa grafičkim dizajnom i za nekoliko minuta kreira željenu publikaciju. Što može biti letak, brošura, čestitka, poslovni formular, pa čak i papirnati avion ili origami. *QuickDemo* prikazuje neiskusnom korisniku kako se obavljeju tipični poslovi u stonom izdavaštvu.

Word Art, opcije za izbor tipa slova i druge efekte u tekstu, podržava sve *TrueType* fontove, dok je celokupna tipografija unapređna. Paket sadrži izuzetne grafičke i dizajnerske alatke, kao što su *Clip Art* (zbirka umetničkih reprodukcija), *Paper Direct* (dizajn papira koji se direktno prikazuje na ekranu), podršku za TWAIN skeniranje i *Kodak Photo CD* tasterski meni se bogatim izborom gotovih slikovnih simbola koji se jednostavno ubacuju u tekst. Osim toga, ovo je prva Microsoft-ove aplikacija koja podržave OLE 2.0.

Microsoft Publisher 2.0 zahteva najmanje DOS 3.1 ili *Windows 3.1*, računar 286, 4 MB memorija, tvrdi disk od 6 – 13 MB, 3.5" HD (1.44 MB) disk-drajv, VGA monitor i miš.

Preporučena cena je 199 USD, ali se do 30. septembra prodaje za samo 139 USD. Korisnicima Microsoft-ovog softvera *Publisher 1.0*, *Word Excel Office* i *Works* se pri kupovini odobrava popust od 20%.

Poslovni softver

Windows u Volstritu: Brokeri sa mišem i kompjuterom

Kompanija „MarketArts" je predstavila softver koji korisniku omogućava da prati tekuće i potencijalne investicije, kreirajući vlastite indikatore i pravile pomoću ugrađene opcija za sastavljanje formula.

Program se zove *Windows on Wallstreet*, radi u okruženju *Microsoft Windows 3.1* i sadrži čitav niz alatki za finansijsku analizu. Opcija za sastavljanje formula omogućava kreiranje preko 5 000 složenih indikatora. *Windows on Wallstreet* je kompatibilan sa programima *MetaStock* i *Computrack*, što mu omogućava analizu „istorijskih" podataka, sakupljenih tokom izvesnog perioda.

Program ima ugrađen sistem za *download* podataka koji sarađuje se velikim brojem izvora podataka, i može da „spušta" podatke simultano dok se na ekranu prikazuje grafikon. Repetativni poslovi mogu da se obavljeju automatsko uz pomoć elektronskog finansijskog pomoćnika – *Personal Investment Assistant* (PIA). On samostalno obavlja i druge poslove (na primer, bekap) i može da se podesi da recimo štampa grafikon samo ako je spuštanje podataka bilo uspešno. To se može obavljati dnevno, sedmično ili jednokratno. Ugrađeni komunikacioni softver automatski obavlja logovanje, prenosenje finansijskih informacija i isključenje. Ili se pak posle logovanja prepušta korisniku da dalje radi sam.

Pomoću *cut-and paste* tehnike grade se formule za izračunavanje najpogodnijeg trenutka za kupovinu ili prodaju deonica. Ugrađena biblioteka ima bogat izbor starijih već godinama korišćenih indikatora, kao i mnoga nova koji su važni za tehničku analizu.

Preporučena cena je 295 dolara.

Korisna adresa: MarketArts, 214-783-6791 ili 800-938-8439, fax 214-783-6790)

ZAMENITE MOTOR AKO SU VAM PODACI SPORI

Odavno au prošla vremana kada ja progrmar pisao sve sam I sve od početka. Danaa se na tržištu može naći obilja biblioteka za najraznovrnija primana, koje u mnogome olakšavaju programarski posao, a u centru naše pažnje, u ovom I nekoliko narednih projeva, naći će se *Paradox Engine*, jedna od najpoznatijih zbirki funkcija za rad sa bazama podataka, i njegova praktična primena u razvoju aplikacija.

Miljan Jovanović i Vladimir Klisić

Pisanje programe zahteve podelu u određene celine koje treba da budu razultat dobre organizacije i ispravnog planiranja projekta. Te celine, u principu, zavise od namene i možde, specifičnosti uslova rada programa. Najčešće se srećemo sa programima koji su sačinjeni od najmanje tri celine: unos/ispis smeštanje i obrada podataka. I dok se (dobre) gotove biblioteke za padajuće menije, prozora i slične stvari mogu pronaći i upotrebiti (utoliko manje posla) sa funkcijama za rad sa bazom podataka to doskore nije bio slučaj. To je jedan od razloga zbog kojih smo posvetili pažnju pojavi najnovije verzije paketa *Paradox Engine*. Ako ste već u prilici da se bavite takvom vrstom posla i ako već treba da izaberete svoj alat onda razmislite o ovom, koji ćemo vam predstaviti sa svim njegovim prednostima i nedostacima. Nadamo se da ćete posle čitanja ovog teksta steći utisak o tome da li je vrađani vašeg truda. Uostalom ponude nije tako velika.

Rad sa bazom je kritičan deo u celom projektu. Kao prvo, baza mora biti pouzdane i otporna *na* razne šokove – nestanak struje i slično, a takođe pristup podacima mora biti jednostavan i brz. Drugo, ako podaci nisu sređeni po nekom kriterijumu, oni nam postaju skoro nedostupni pa se kao jedan od osnovnih zahteva postavlja i sortiranje podataka. Mehanizam ovih funkcija more da pruži i mnoge druge mogućnosti, koje će biti opisane u ovom tekštu.

Kako je opsežan prikaz ovog paketa iz ugle organizacije i mogućnosti već našao mesto na stranicama ovog časopise mi se nećemo zadržavati *na* njemu. Sloge prediažemo da, pra nego što se upustite u čitanje ovog pročitate tekst Dušana Popovića, objavljen u martu ove godine (R90/26).

PRAKTIČNI PROBLEMI

Paradox Engine se isporučuje kao biblioteke funkcije za rad sa bazom podataka. Podržani su *C* i *Pascal* u *DOS* i *Windows* okruženju ali će predmet našeg interesovanja biti samo *C* verzija za *DOS* jer se kroz njen prikaz sesvim lepo mogu izneti opšti principi na kojima radi *Paradox Engine*. Tekst će otud biti prevashodno namenjen *C* programerima i onima koji se tako osećaju. Takođe podrazumevaće se rad se *Borland C* prevodiocem (BCC), meda se on u osnovi uopšte ne razlikuje od rada se *Microsoft C* (MSC) prevodiocem – razlike postoje samo na nivou pravođenja e ne i upotrebe funkcija. Naravno, u takstu će na ove razlike biti ukazano.

Da bi vaš program koristio funkcije *Paradox Engine*-a obavezno morale uraditi dve stvari prva je da negde na početku vašeg programe #include direktivom uključite u proces prevođenja zaglavlje **PxEngine.h** jer su u njemu sve neophodne informacije za pravilan rad *Engine*-a – prototipovi funkcije konstante itd.

Drugi korak je da prilkom povezivanje u vaš program uključite odgovarajuću *Paradox Engine* biblioteku. Koja će to biblioteka biti, zavisi od prevodioca koji konstrite ze BCC je potrebna biblioteka **PXENGTCL.LIB** a ze MSC biblioteka **PXENGMSL LIB**. U oba slučaja treba da obratite pazniju na činjenicu da su biblioteke isporučene u *LARGE* memorijskom modelu, pa se tome mora prilagoditi i vaš program. Da vidimo kako.

Kada je reč o BCC prevodiocu na raspolaganju su vam dva načina rada pri prevođenju se komandne linije treba navesti **-ml** prekidač i na kraju rede dodati **PXENGTCL.LIB** dok ze rad u integrisanoj razvojnoj okolini (IDE) treba iz menije **Options\Compiler\Code generation** izebrati **Large** opciju i u projektnu datoteku (.PRJ) uključiti **PXENGTCL.LIB**. Podržan je i rad Borlandove **VROOMM** tehnologije pa se prekidač **-Y** može slobodno koristiti. U tom slučaju bi komandna linija izgledale otprilike ovako

```
bcc -ml -Y <dat.c> . -Yo PXMSG.C
PXENGTCL.LIB
```

Da bi se isti program preveo MSC prevodiocem potrebno je uraditi sledeće

- Navesti gde se nalazi zaglavlje pxengine.h (**SET INCLUDE**= <**path**>;**%INCLUDE%**)
- Navesti gde se nalazi biblioteke pxengmsl.lib (**SET LIB**= <**path**>;**%LIB%**)
- Pozvati MSC prevodilac komandom

```
CL /AL /F D00 [opcije] <dat.c> PXMSG C
[-link opcije] PXENGMSL.LIB
```

/AL	- large model
/F D00	- povećava podrazumevenu vrednost steka
<dat.c>	- program koji se prevodi
PXMSG C	- modul sa oplsima grešaka i funkcijom ze njihovu obradu
[-link opcije]	opcije linkera
PXENGMSL LIB	- *Paradox Engine* biblioteka se funkcijama za rad se MSC prevodiocem

Verovatno ste primetil da je modul **PXMSG.C** u kojem je funkcija za ispis grešaka, izdvojen iz biblioteke i ostavljen u izvornom obliku. Borlandovi programeri su imali dobar razlog za to: time su vam dali potpunu slobodu da prepravljate posliječu ili čak napišete novu funkciju za obradu grešaka koje će recimo sve poruke ispisivati na našem jeziku. Jednom kada prepravite taj modul možete ga prevesti i dodati u biblioteku – isto nećete morati da ga navodite svaki put ne komandnoj linji

```
BCC -c -ml pxmag c
TLIB /C pxengtcl lib +pxmsg.obj
```

Nakon što smo rešili praktična problemе oko prevođenje, podignimo haubu i pogledajmo sem motor.

DATOTEKE

U toku rada *Paradox Engine* koristi veći broj različitih datoteka da bismo videli koja datoteka čemu služi, treba da razmotrimo organizaciju podatake na disku. *Paradox Engine* podatke smešta u tabele, a one opet mogu imati objekte koji su im pridruženi tako da čine jednu celinu – **familiju**. Promene obavljene na jednoj tabeli u opšlem slučaju izazivaju odgovarajuće promena članove njene familije. Na disku, datoteke jedne familije imaju isto ime ali se razlikuju po tipu

DB	- tabela
PX	- primarni indeks
X??, Y??	- sekundarni indeks (?? = 01-FF) mela i velike slova se razlikuju (case sensitive)
XG? YG?	- sekundarni indeks (? = 0-F) mele i velike slova se NE razlikuju (case insensitive) ili kom pozitni indeks (sastavljen od više polja)
MB	- BLOB (Binary Large OBject – ekvivalent memo polju samo moćniji)

Tabela je izvedena kao niz slogova (**records**) koji se sestoje od polja (**fields**). Broj slogova u tabeli je ograničen podrazumevanom maksimalnom dužinom tabele (256Mb) ili, što je verovatnije, slobodnim prostorom na disku. Maksimalni broj polja u slogu je ograničen *na* 255 ili dužinom sloge od 4000 bajtova (za indeksirane tabele 1350). *Paradox Engine* uvodi neke nove tipove indeksa, pa se pozabavimo najpre njima.

PRIMARNI I ...

Indeksne datoteke služe za ubrzavanje pristupa slogu u tabeli. *Paradox Engine* može otvoriti više vrste indeksa **primarne sekundarne sekundarne sa ažuriranjem** i **kompozitne**. Tekode indeksi mogu biti kreirani tako da se vodi računa o razlici između malih i velikih slova. Ako je potrebno indeksi se kreiraju *pre* otvaranja tabele a otvaraju se prilikom otvaranja tabele.

Primarni indeksi zahvetaju prvih *n* polje posmatrano sleva nadesno. Ako treba formirati primarni indeks po trećem polju, onda se njime moraju zahvatiti i prve dve.

slog polje1 polje2 polje3 polje4

primarni indeks može da bude

- **po polju 1**
- **po poljima 1 i 2**
- **po poljima 1, 2 i 3**

Nikako se ne mogu presakati polje ili menjati redosled. Primarni indeks je male dužine i može se reći da ne opterećuje prostor na disku (npr. tabela od 100 000 slogova dužine oko 3 9Mb ima primarni indeks dužine oko 55kb!). Ovo je vrlo bitno jer otvaranje tabele po nekom drugom indeksu zahteva poslojanje primarnog indeksa.

... SEKUNDARNI ...

Sekundarni indeksi se koriste za sortiranje tabela po nekom drugom polju (koloni), i za njihovu upotrebu neophodno je da postoji primarni indeks. Postoje dve vrste sekundarnih indeksa. Oni na koje se odražave svaka promena tabele (**INCSECONDARY**) i oni koji su posle promene tabele neupotrebljivi (**SECONDARY**). I dok je za **INCSECONDARY** indekse jasno gde se mogu upotrebiti **SECONDARY** indeksi nemaju smisla. Ipak postoje situacije kada se i oni mogu upotrebiti: ako je u nekom trenutku potrebno izvršiti neku operaciju nad svim (ili većim brojem) slogova u tabeli pogodno je za tu priliku otvoriti takav indeks koji će po obavljanom poslu biti obrisan. Naravno, treba imati u vidu da ovakva intervencija traži vreme i prilično prostor na disku, jer sekundarni indeksi (obično) zauzimaju više prostora na disku nego sama tabela (što nije baš pohvalno!). Opet, i u ovim indeksima se može voditi računa o razlici između malih i velikih slova:

... I KOMPOZITNI INDEKS

Kompozitni indeksi su novitet verzije 3.0. Oni zapravo omogućavaju indeksiranje po proizvoljnom rasporedu polja. I za njih su primenjivi indeksi obavezni. Indeksiranje kompozitnim indeksom se izvodi iz dva koraka:

- Kreira se logičko polje od niza polja po kojima treba napraviti indeks. Za to logičko polje (koje ima i ime) dodija se **id** broj baš kao i za svako drugo fizičko polje.
- Daji postupak je identičan kreiranju indeksa po jednom polju.

Ono što se ovde može primetiti je da se ne mogu kreirati indeksi po konsničkoj funkciji- što je, recimo, izvodljivo u *Clipper*-u. Takođe ni kodni rasporedi koji su nam na raspolaganju za sortiranje ne idu nam na ruku. Činjenica je da postoji nekoliko različitih kodnih rasporeda po kojima se može obaviti sortiranje- ali je tu *Paradox Engine* potpuno zatvoren za izmena, pa se problem sortiranja po našim slovima čini nerešivim (hm, videćemo...). Podržani su rasporedi za razne govorna područja:

SortOrderAscII (default)
brojevi, velika slova, mala slova, akcentovana slova

SortOrderIntl
brojevi, slova (nema razlika između velikih malih i akcentovanih slova)

SortOrderNorDan (iz verzije **Paradox 3.5**)

SortOrderNorDan4 (iz verzija **Paradox 4.0**)

SortOrderSwedFin
svi identični **SortOrderIntl** rasporedu s tim što je izmenjen slova **z** i niz njihovih slova

PRINCIP RADA

Da bismo opisali način na koji *Paradox Engine* funkcije obavljaju svoj posao treba da se upoznamo sa konceptom slog-bafera (**Record Buffer**) koji predstavlja suštinu čitavog mehanizma. Naime, prenos podataka između programa i tabela *ne* diska obavlja se upravo preko ovih bafera. Sve što se upisuje na disk ili čita sa njega mora proći kroz njih, jer je to jedini način da *Paradox Engine* proveri ispravnost podataka (njihov tip, opseg i dužinu). Ovi slog bafer nisu ništa drugo do niz ovih bajtova u memoriji dugački upravo onoliko koliko i slog u tabeli za koju se otvaraju. Kad kažemo otvaraju, to znači da nam nisu odmah na raspolaganju podrazumeva se da ih prvo moramo otvoriti, i tek tada se fizički odvaja memorije za njih. *Paradox Engine* svakom otvorenom slog-baferu dodeljuje identifikacioni broj (**RECORDNANDLE**) koji je tipa **unsigned int**. Kasnija kada se prostupa nekom slog baferu navodi se samo njegov **id**. Koliko se slog bafera može otvoriti istovremeno zavisi od slobodne memorije, ali i od uslova pod kojima je inicijalizovan *Paradox Engine*. **Id** je globalna stvar – ako je otvorano 5 tabela i uz svaku po tri bafera, onda će oni imati **redom vrednosti od 1 do 15**. To znači da u svakom trenutku imate prislup svakom baferu – kako onima koje ste otvorili uz tabelu na kojoj trenutno radite tako i svim drugim. Teorijski možete u naku tabelu upisati čak i podatke iz slog-bafera koji nije otvoren za nju naravno, da bi vam to uspelo potrebno je da laj slog-bafer ima strukturu i veličinu identičnu slogu u tabeli. Drugo pitanje je, međutim, da li to ima smisla u praksi. Sa druge strane, transfer između slog-bafera iste tabele ima smisla, i za njega je obezbađena odgovarajuća *Paradox Engine* funkcija.

Zanimljivo je pitanje koliko nam je bafera po tabeli dovoljno u praksi. Naše iskustvo bi se moglo sažeti u jednu reč: tri. Jedan bi mogao poslužiti kao radni bafer. I u njemu bi se vršila izmena, drugi bafer bi služio za privremeno čuvanje starog sadržaja radnog bafera, recimo za slučaj da upis novih podataka na disk ne uspe, treći bi služio kao bafer za preuređivanje tabele (o tome kasnije). Naravno, sve se (ili gotovo sve) može izvesti i sa jednim slog-baferom po tabeli – vaša je stvar koliko ćete ih otvoriti.

Slog-bafer se popunjava tako što se popunjavaju njegova polja. Analogno se čita njegov sadržaj. Princip po kojem se prostupa slog-bafernima identično je primenjen i na polja unutar njih: svako polje posebnom funkcijom dobija svoj identifikacioni broj (**FIELDNANDLE**) tipa **unsigned int**, pomoću kojeg mu se kasnije pristupa. Ovo zeli-ste daluje nezgrapno – zamislite deset polja u slog-baleru za koje treba deset puta pozvati funkciju za dodelu **id** broja i kasnije... desal poziva funkcija za čitanje svakog polja pojedinačno ili deset poziva funkcija za pisanje. Sve u svemu mukotrpan posao. Neka stvari se, doduše, mogu i zaobići: **id** brojevi polja idu uvek u rastućem redosledu od jedinice (za krajnje levo polje) pa se funkcija za dodelu id broja i ne mora pozivati. Međutim, još uvek ostaju oni pozivi za čitanja i pisanje. Pretpostavljamo da su ovo bile glavne primedbe na verziju 2.0, pa se u verziji 3.0 pojavljuju (spasonosne?) rešenja ali o tome u nekom od nastavaka. Da rezimiramo: kada se slog čita iz tabele, njegov sadržaj se kompletno prepisuje u slog-bafer gde nam stoji na raspolaganju za izmene. Da bi se slog upisao u tabelu potrebno je popuniti polja slog-bafera otvorenog uz tabelu i potom čitav njegov sadržaj preneti u tabelu.

Kako će *Paradox Engine* znati u koju tabelu treba upisati sadržaj slog-bafera? Vrlo prosto: i tabela prilikom otvaranja dobija svoj identifikacioni broj (**TABLENANDLE**) opet tipa **unsigned int** i kasnije se pomoću tog broja vrši referisanja na nju.

Ovo je suština reda sa *Paradox Engine*-om kada sa naviknete na ovakav način razmišljanje stvar je vremena naučiti koja se funkcija kako zove i sa kojim se parametrima poziva. Naravno, da bi se napisao prvi ozbiljan program koji koristi ove funkcije potrebno je znati još puno toga o njima pa zato pogledajmo šta nam još stoji na raspolaganju.

PARADOX FUNKCIJE

Imena svih *Paradox Engine* funkcija počinju velikim slovima *PX* i sačinjena su od nekoliko reči tako da se iz njih može zaključiti na koji se objekat odnose i koju operaciju nad njim obavljaju. Tako se može izvesti podela na sledeće grupe:

Ime	operacije
PXTbl	nad tabelama
PXRec	nad slogovima u tabeli
PXBlob	nad BLOBovima
PXRecBuf	nad slog-balerima
PXFld	nad poljima
PXNet	mrežne funkcije
PXPew	šifrovanja i zaštita tabela
PXDate	nad datumima
PXKey	nad indeksima
PXPut	punjenje polja slog-bafera
PXGet	čitanja polja slog-bafera
PXSrch	pretraživanje
PXRaw	rad sa slog-baferom u celini

Tri tačke sloje za konkretan opis neke operacije: **Open, Close, Add, Update, Init**, itd.

PROMENA MAKSIMALNOG BROJA OTVORENIH DATOTEKA

Kao prvo, namojte zabotraviti da promenite liniju u **CONFIG.SYS** datoteci ali ovo neće biti dovoljno. Potrebne su još izvesna izmena u standardnoj biblioteci **Large** modela (ako vam je potrebno, isti se postupak može obaviti i nad ostalim modelima), a ona se sastoje iz sledećeg:

1) editovati datoteku **INCLUDE_nfile.h** i u njoj podesiti red

#define _NFILE_ 20

tako da sadrži potreban broj (npr 60)
2) prevesti dva datoteka koje sa nalaze u **LIB** direktorijumu

bcc -c -ml files.c files2.c

3) objekte **files.obj files2.obj** uključiti u **cl.lib**

tlib cl.lib +-files.obj +-files2.obj

Ovim je biblioteka **cl.lib** spremna za upotrebu i na dalje će vam biti dozvoljeno otvaranje 60 datoteka.

Datoteka **files.c** sadrži niz struktura **FILE** i fu se razavriše mesto za **id** datoteka otvorenih preko funkcija **fopen()** i vrši se inicijalizacija prvih pet članova na **stdin stdout stderr, stdaux** i **stdprn**. Datoteka **files2.c** poređ sličnе definicije za otvaranja datoteka preko funkcija **open()** sadrži i red

unsigned _nfile = _NFILE_;

koji zapravo dafiniše promenljivu **_nfila** korišćenu u **startup** kodu (c0.asm) gde se zapravo vrši podešavanja maksimalnog broja fajlova.

Napomena: Pretpostavljam da će postojati izvesni problemi ako se Vaša aplikacija na buda konstila pod dosom 3.30 ili novijim. O ovoma možete više pronaći u **C0.ASM**, u **LIB\STARTUP** direktorijumu.

Opisane promene se odnose na BC++ 3.x, ali ne znam da li se ovo može uraditi i u verziji BC++ 2.0.

Znanje o tome šta se sve može učiniti sa *Paradox Engine* funkcijama nije dovoljno potrebno je znati i *kako* ih i što je ne manje bitno, *kada* upotrebiti. U svakom programu gde se koristi *Paradox Engine* funkcije mora se najpre obaviti inicijalizacija *Paradox Engine* okruženja a po završetku rada vratiti sistem u prvobitno stanje. Evo kako

PALJENJE MAŠINE

Po analogiji sa BGI standardom *Paradox Engine* se "pali" pomoću **Init** funkcije. Ako je reč o aplikaciji koja se izvršava na samo jednom računaru biće to funkcija **PXInit()** u mrežnom okruženju funkcija **PXNetInit()** (logično?) i najzad, pod **Windows**-ima funkcija **PXWinInit()**. U prvom slučaju nije potrebno navoditi nikakve argumente a ostali slučajevi biće obrađeni detaljno u nekom od tekstova koji slede u ovoj seriji.

Šta se događa prilikom inicijalizacije *Engine* okruženja? Odvaja se memorija za interne promenljive bafere i strukture i vrši se njihovo postavljanje na početne vrednosti. Prilično je važno znati čime u startu raspolažemo koliko tabela možemo istovremeno držati otvorenim, koliko slog-bafera imamo koliko je veliki interni bafer koji služi za ubrzavanje rada sa diskom. Sve se to može zadati funkcijom **PXSetDefaults()** koja jasno mora biti pozvana *pre* funkcije koje vrši inicijalizaciju. Zadržimo se malo detaljnije na njoj.

Prototip ove funkcije izgleda ovako

```
int PXSetDefaults(int bufSize, int maxTables, int maxRecBufs, int maxLocks, int maxFiles, char *sortOrder);
```

bufSize veličina internog bafera za transfer na disk. Njegova svrha je da ubrza rad sa diskom. To praktično znači da se posle poziva funkcija koje upisuju slogove na disk podaci ne nalaze odmah tamo gde ih očekujete, u tabelama već u tom internom baferu. Imajte to u vidu kada vršite pretraživanja. Dozvoljene vrednosti ovog bafera su između 8 i 256k, a podrazumevana vrednost je 32k. Veće vrednosti su mač sa dve oštrice: rad može biti ubrzan ali i šteta ako nestane struje (osim ako nemate **UPS**). Treba proveriti da li se zaista radi o ubrzanju jer iskustva govore da se velikim baferom, radi čitanja kraćih podataka, može značajno usporiti rad se datotekama. Ovo iskustvo nije vezano samo za *Paradox Engine* već uopšte. (Radi čitanje nekoliko bajtova u određenim situacijama, puni se ceo bafer.) Postoji i mogućnost da upis ide direktno na disk i tada su vaši podaci najsigurniji. Odluka o načinu rada sa diskom donosi se za sveku tabelu posebno. Prilikom njenog otvaranja a postoje i situacije kada se podaci obavezno odmah snimaju na disk (u mrežnom okruženju, pod određenim uslovima)

maxTables broj tabela koje mogu biti istovremeno otvorene. Ovaj broj se kreće u raponu od 1 do 64; a inicijalno je 5. Ako vaš program prijavi grešku **No more table handles available** znajte da je razlog tome mala vrednost ovog broja.

maxRecBufs broj slog-bafera koji istovremeno mogu biti otvoreni. Dozvoljene vrednosti su između 1 i 512 a podrazumevana vrednost je 10. Keko je već napomenuto baferi nemaju istu dužinu: svaki bafer će zavisno od tabele uz koju je otvoren imati dužinu jednaku dužini sloga te tabele. Opet više slog bafera znači veću potrošnju memorije.

maxLocks broj zaključavanja koja se mogu primeniti na jednu tabelu istovremeno. Ovaj broj se kreće između 1 i 128 a početno je postavljen na 32. Ovaj broj ima svoj praktični značaj samo pri radu u višekorisničkom okruženju.

maxFiles broj datoteka koje *Engine* može da otvori. Podrazumevana vrednost je 10. Da bi se pod *Engine*-om otvorilo recimo više od 20 datoteka *nije* dovoljno da ovaj broj bude postavljen u skladu sa time. Da bi to bilo moguće potrebno je izvesti neke zahvale na **clib** (**LARGE** model standardne biblioteke) rešenje ovog problema je dato na slici 1, za **Borladov C** prevodilac 3.x). Ako vam se čini da su retke situacije kada se otvare više od 20 datoteka istovremeno, mali račun će vas uveriti da to nije redak slučaj, pored standardnih 5, sam *Engine* otvara još nekoliko zatim se otvaraju tabele i indeksne datoteke, pa eko još imate i datoteke u koje upisujete poruke o sistemskim greškama.

sortOrder kodni raspored po kojem će se vršiti sortiranja. Ako bilo koji od ovih para metara ne želite da menjate dovoljno je da u pozivu funkcije **PXSetDefaults()** na njegovom mestu stavite **PXDEFAULT** (konstanta, vrednosti 0L) i *Engine* okruženje će biti inicijalizovano sa njegovom podrazumevanom vrednošću.

Napomena neke od navedenih promena slu paju na snagu tek kada se aktivira odgovaraju ća **PX** funkcija. To znači da pri inicijalizaciji *Engine*-a **nece** doći do stvarnog alociranja potrebne memorije već će se samo interno zapamtiti vaš zahtev i tek u odgovarajućoj prilici proveriti i obraditi. **Primer** sve bok negde u programu ne navedete da želite otvaranje slog-bafera memorija za **mexRecBuf** broj slog-bafera neće biti odvojena. Tada će se proveriti ima li dovoljno memorije i u slučaju da ima, vaš program će normalno nastaviti rad. Dakle *Engine* (u ovom slučaju **PXSetDefaults**) nema načine da sazna ime li u startu dovoljno memorije za vaše potrebe. Nije ni potrebno. Ovo je ekonomičnija varijanta i ze ves nepovoljnije, jer je na vama da vodite recune o tome imate li dovoljno proslora na disku ili dovoljno slobodne memorije – ali uostalom zar u C-u nije uvek tako?

Ako inicijalizacija uspe vaš program nesmetano može da poziva one funkcije koje su omogućene tipom inicijalizacije (*Windows* meža). Veoma je važno *uvek* proveriti da li je *Paradox Engine* funkcija obavila svoj posao. To se odnosi na sve funkcije. Ignorisanje ili neproveravanje onoga što vraćaju *Paradox Engine* funkcije prevrolinjski vodi *Engine* u nestabilno stanje a vaš program u kreh. Greške i njihova obrada zahtevaju posebnu pažnju, a kako je ovo prvi članak u seriji i uz to na ograničenom prostoru, posvetićemu im pažnju (pogađate) u nastavcima koji slede.

Po završetku rada potrebno je 'pospremiti' sven za sobom ispraznti bafera (promene se snimaju na disk) oslobediti zeuzelu memoriju zatvoriti datoteke otvorane preko *Paradox Engine* funkcija otključati eventualno zaključane datoteke i obnseti privremene datoteke. Taj posao je poveren funkciji **PXExit()**

Najkreći program koji se sastoji samo od poziva funkcije **PXInit()** i **PXExit()** ima otprilike 160kb. To je zalo što su u program odmah uključeno sve **PX** funkcije. Mede vam dužine programa može smelati ipak ne računajte na lo da možete izdvojiti iz biblioteke neke od modula za koje pouzdano znate da vam neće trebati – nikad ne znate koja **PX** funkcija ih može mlerno pozivati.

Toliko za početak. Ovo je ipek samo uvod – znate kako se kaže. *The best is yet to come!*

(Na Sezamu mjove i klise)

STA IMA NOVO

SOFTVER

Obrada teksta

Deset godina MS WORD-a: O slavljenicu sve najlepše

Microsoft Word se prvi put pojavio pre deset godina. Godinu dana kasnije izašle je verzija za Macintosh a 1989. za Windows okruženje. Tokom poslednjih nekoliko godina masovni prelazak na *Microsoft Windows* kao i sve veća potreba za među-platformskom kompatibilnošću podstakli su prodaju *Worda*. Ubzajen faktor je i rastuća popularnost paketa *Microsoft Office* koji objedinjuje *Microsoft Word Microsoft Excel* program za radne tabele *PowerPoint*, program za grafičke prezentacije kao i licencu zaCMDNMC radnu stanicu *Microsoft Mail*. Kažu da se gotovo 50% primeraka *Word*-a proda u paketu *Microsoft Office*.

O slavljeniku naravno ove najlepše naročito iz usta njegovog tvorca, pa se tako ističe da je *Word* bio prvi tekst procesor sa podiškom za miša (1983 *Word 1.0* za MS DOS) prvi je podržao laserske štampače (1986 *Word 3.0* ze MS-DOS za mreže) prvi koji je ponudio „drag-and-drop" funkciju (1991 *Word 2.0* za *Windows*) itd.

Microsoft ne propušta da naglasi činjenicu da je gotovo milion korisnika prešlo sa *WordPerfect*-a za MS DOS na *Word* za *Windows* da je prodaja za poslednje tri godine utrostručena i da je *Word* dobio pedesetak priznanja od vodećih kompjuterskih publikacija. Osim toga, najčuvenije visoke škole za biznismena na univerzitetima Stanford Harvard i Jejl kao i mnoge od najprestižnijih advokatskih firmi su za svoj tekst-processi izebrale upravo *Microsoft Word*.

Uspeh koji *Word* ima širom sveta u velikoj meri se pripisuje činjenici da se trenutno može na 25 jezika, za MS DOS Macintosh ili *Windows* okruženje što omogućava jednostavnu razmenu dokumenata ne različitim jezicima. Između različitih platformi *Word* prodira i na nova tržišta – nedavno su izašle verzija na japanskom (Kanji) tajlandskom i turskom a uskoro se očekuju na arepskom mađarskom, ruskom i poljskom.

Sve to nije dovoljno ambicioznim *realizmima* te desetogodišnjaku predstoje nove i nova usavršavanja. Sledeća izdanja će dobiti „magičnu" funkciju (tekozvani *wizard* – čarobnjak ili mag) za ubrzan i kraj nije istančan dizajn tekstka kao i unapređene OLE funkcije. Ulože se i u dalekosežnije tehnologije kao što su prirodni jezik i druge oblici primenjene inteligencije koje će obradi teksta dati sasvim novu dimenziju.

Prepoznavanje teksta

Optički čitači

Sve veća upotreba softvera za optičku prepoznavanje grafičkih znakove (OCR) dolazi do izražaja i sa pojavom novog „Logitech"-ovog paketa krekranog za ScanMan poroduicu *Windows* ručnih skenera.

OmniPage Direct AnyFont OCR softver koriste *Casere AnyFont* mašinu *AnyPage* tehnologija i OCR alatku koje po rečima proizvođača omogućava korisniku da skenira i pročita tekst praktično iz svih *Windows* aplikacije.

Softver je trebalo da se pojavi u prodaji tokom jula a već se nude razne dogradnje za stare korisnike „Logitech"-ovih proizvoda – *CatchWord Pro Perceive Personnl CatchWord DOS* ili bilo koje *ScanMan* verzije. Predstavnik firme kaže da korisnici sva više traže pouzdano i kompletno (podjednako precizno i za slike i za tekst) rešenje za ručne skenere koje će ipak ime ti razumnu cenu.

OmniPage Direct AnyFont OCR softver za „Logitech" nudi snagu *Casre* tehnologije i koristan niz osnovnih alatki dok proveru pravopisa i gramatike *cut-and paste* i druge editorske poslove ostavlje desti nacionoj aplikaciji. Tako se dobije visoko-funkcionalan proizvod krajnje lak za korišćenje. Kažu da će 70-ak aplikacija za obradu teksta stono izdevaštvo i račine tabele u svojim paketima ubuduće sadržati i ovaj softver.

Potreban je 386 ili bolji računar, 4 MB RAM i *swap* datoteka od 4 MB. DOS 3.1 ili novija verzija. *Windows 3.1. Logitech ili drugi Windows* kompatibilan miš i Logitech ručni skener.

NAJVEĆI IZBOR LICENCNOG SOFTVERA

Borland Turbo Pascal 7.0 • Borland Pascal with objects 7.0 • Borland C++ 3.1 with application frameworks • Borland C++ 3.1 • Paradox 4.0 • Paradox Engine 3.0 and database frameworks • Quattro Pro 4.0 for DOS • Framework IV • Brief 3.1 • DBase IV V2.0

Microsoft MS DOS 6 upgrade • Windows 3.1 english • Windows 3.1 Eastern Europe • Windows 3.1 for Workgroups • Word for Windows 2.0 • Excel for Windows 4.0 • Works for Windows 2.0 English • Word for DOS 5.5 • Visual Basic for Windows 2.0 standard • Visual C++ 1.0 standard • FoxPro 2.5 for Windows

Lotus 123 Ver 3.4 • 123 for Windows • Ami Pro 3.0 • Lotus Write • Freelance/DOS • Magellan 2.0 • Agenda 2.0 • Lotus Works 1.0 • Symphony 3.0

Utilities Norton Antivirus 2.1 • Norton Backup 1.2 • [illegible] • Norton Backup for Windows 1.2 • PC Tools 8.0 • PC Tools for Windows

CAD/CAM Autodesk AutoCAD 386 R12 / DOS • Autodesk 3D Studio 2.0 & CD • DesignCAD 2D 5.0 • DesignCAD 3D 3.1

SPRINT
INFORMACIONI SISTEMI

DTP/GRAPHICS Adobe Type Manager 2.0 • Adobe Illustrator • PageMaker 5.0 • CorelDRAW! 3.0 & CD • Ventura for Windows 4.1 • QuarkXPress for Windows • Harvard Graphics 3.0 • Photo Styler • Photo finish 1.0 • Publisher's Paintbrush 2.0

WORDPROCESSORS/DATABASE WordPerfect 5.1 • Wordstar 2000 3.5 • Wordstar 7.0 • Clarion 2.1 Professional • Clipper 5.2 • Superbase V 2.0

PROGRESS DATABASE PRODUCTS Progress Application Development System (DOS or Windows) • Query/Report (DOS or Windows) • Progress Run-Time • Developers Toolkit • Database Gateways...

PROGRAMMING TOOLS CodeBase++ • Zinc 3.0 for DOS and Windows • Novell Btrieve for Windows • Win++ for Windows

COMMUNICATIONS [illegible] LapLink Pro 4.0 • Procomm Plus/win • Procomm Plus 2.0 • Carbon Copy Plus 6.0 (double)

NOVELL NETWARE Netware 4.0 • Netware 3.11 • SFT III • Netware 2.2 • Netware Lite...

NOVI SAD • APOLO CENTAR • POSLOVNI PROSTOR 59 • III SPRAT • TEL. 021/611-502

BOOM

VAŠA SINGAPURSKA VEZA

2.000 komada **on stock**

ProVitek
14" Super VGA
Color Monitori

TOWER KUĆIŠTA
MINI

INTERSOFT MICROCOMPUTERS

Bgd. 29. novembra 43/1
3221 959 3223-539
3224-120 3225-393

CIM - STRATEGIJA PREDUZEĆA

CIM - Computer Integrated Manufacturing strategija je novi koncept vođenja preduzeća. U svetu se ta strategija ili filosofija primenjuje od 1985. godine i ista obezbeđuje bolju poziciju preduzeća na tržištu u uslovima oštre tržišne konkurencije. Ta strategija polazi od marketinga i ima organizovanu strukturu obezbeđuje delijanje proizvoda koji zadovoljava zahteve kupca, a da pri tome nastaju minimalni troškovi za definisani kvalitet. Jasno je da je za realizaciju takve strategije neophodno da se koriste "novi alati". Ti "alati" su ugrađeni u sledeće module CIM-a: CAD - projektovanje je uz veliku primenu računara, CAP - primerne podrške računarom, CAM - proizvodnja podržana računarom, CAQ - kvalitetna podrška računarom, PPS - planiranja i upravljanja proizvodnjom i MIS - menadžment informacionog sistema.

Danas nema ni jednog značajnog proizvođača u svetu koji ne koristi predmeti CIM strategije. To je poznato i našim menadžerima i stručnjacima. Ne želeći informacije su kojima u ljudi raspolaži su nedovoljne za pravilno definisanje svojih CIM ciljeva, za definisanje CIM projektnog zadatka, kao i za izbor pravilne strategije planiranja i uvođenja CIM koncepta. Iz tih razloga dolazi do velikih lutanja i finansijskih promašaja, koji najčešće kompromituju ideju uvođenja CIM koncepta u preduzeće.

NARUDŽBENICA

Naručujem knjigu CIM STRATEGIJA PREDUZEĆA

Za cenu knjige pitati na tel 011/ 711 725
CIM COLLEGE, Milana Blagojevića 30, 1000 Ns

Broj primeraka: _____
Ime i prezime (naziv firme): _____
Poštanski broj i mesto: _____
Ulica i broj: _____
Telefon/ Telefaks: _____

M.P. Potpis

Da bi se izbegla nepotrebna lutanja, finansijski promašaji i da bi stručnjaci i menadžeri imali prave informacije o značaju i sadržaju CIM pojma, neophodno je dodatno obrazovanje ili kadrova. To dodatno obrazovanje je moguće kroz seminare inovacije znanja, kao i praćenjem literature iz te oblasti. Na našem tržištu do sada se nije pojavila ni jedna knjiga koja bi obrađivala razmatranu problematiku. Zato knjiga CIM STRATEGIJA PREDUZEĆA treba da postane izvor potrebnih informacija kako za menadžere i stručnjake o privredi tako i za savremenije školovanje studenata na našim univerzitetima.

Knjiga CIM STRATEGIJA PREDUZEĆA ima 400 stranica na 247 slika i podeljena je u 9 glava.

STROGO PO KALENDARU

Da li vam ja nekada zatrebalo da odredite u kojl je dan pao neki događaj, koliko je vremena proteklo između dva datuma, kada će tačno isteći prvih 100 dana neka vlade ili, uopšte, da obavite neku kalendarsku čarolíju?

Dejan Ristanović

Ako pišete poslovni softver sigurno jeste a sigurno je i da ste se snašli i napisali neki manje III više složan program koji je „obavio posao". Ovim tekstom ćemo pokušati da „za avagda" rešimo probleme koji se javljaju pri radu sa datumima, da predstavimo potprograma koje ćete docnije moći da ugrađujete u vaša remek dela, uz sigurnost da će korektno tretirati sva datume koje im „podmetnete" i, što će možda mnoge zanimati da objasnimo **kako** sve to radi! I **zašto** je baš tako!

Kada bismo imali dobar kalendar, pitanja kojima se bevi ovaj takst se na bi ni postavijala – svaki bi mesec imao jednak broj dana, prvi u mesecu bi iz godina u godinu padao na isti datum pa bi malo pamćenja rešilo sve problema Međutim kalendar je takav kakav je pa su se pitanja tipa „u koji je dan pao 19 novembar 1825. godine" i te kako postavljala i rešavala raznim metodama Ti metodi su se uglavnom zasnivali na složenim tabelarne koje su objavljivane u knjizi zvenu „Večiti kalendar" („večnost" se obično tamo svodila na intervet od sto ili dvesta godina) koja je dopunjavana sanovnikom, horoskopom rođendnikom I drogim glupostima tog tipa Problem konverzije datuma mi je „konačno" rešio moj prvi računar (bolje reći programabilni kalkulator) TI-58 – prisećam se da je u njegovu standardnu bibliotaku (*Master Library Module*) ugrađen program pomoću kojeg se odreduje u koji dan neki datum pada I koliko je dana proteklo između dve datuma Naprosto otkucate 1119 1825, izaberete PGM 20 pritisnete taster na kojem piše alovo D, sačekate trenutak i ato dana!

Znatno bolji program koji obavlja isti posao dobio sam u svojim HP-41C danima, pošto sam kupio ROM amerčkog PPC kluba – u njega su ugrađeni programi CJ I JC koji konvartuju datum u referentni broj, ali i referentni broj u dan mesec i godinu Programi su, kao uostalom i čitav PPC ROM prilvo malo čudo optimizacije i godinama su mi zamenjivali kalendar – i dan danas ću, ako mi treba da proverim u koji dan naki datum pada, uključiti HP-41CV U međuvremenu sam pregledao brojne biblioteke matematičkih i poslovnih funkcija raznih računara i kompajlere i pronašao najrazličitije algoritme za rešavanje problema – ima ih boljih i gonh pouzdanijih i menje pouzdanih, sporijih i bržih u svoje programa već godinama ugrađujem iste procedure koje sam, oglavnom po ugledu na PPC ROM napisao još davna 1985 godine, najpre na bejziku a docnije ne paskalu C-u *Clipper*-u De bi se ova procedure kreativno primenile potrebna je naravno, određena teorijska podloga koju ćemo ovda pokušati da izložimo

REFERENTNI BROJ DATUMA

Red sa datumime se zapravo zasniva na dva osnovne operecija Prva je pretvaranja datuma (zadatog u obliku datum/mesec/godine) u referentni broj tj izračunavanje broja dana koji su protekli između nekog nultog trenutke u istoriji i zadatog datume Druga operacija je inverzna – za zadati referentni broj treba odredit datum, mesec i godinu koji mu odgovaraju Recimo da ja nulti trenutak istonje 1 septembar 1993 godina i „posmatrajmo" neka realne datume Referantni broj samog 1 septembra 1993 biće 0, referentni broj 2 septembra 1993 će biti 1 a, recimo, 15 septembra 1993 biće 14 Sada nas zanima koji će datum biti 10 dana posle 15 septembra 1993 Izračunaćemo referentni broj 15 septembra (14), zatim mu dodati 10 (24) i onda za taj referentni broj izračunati datum mesec i godinu – dobićemo 25 septembar 1993 Što se dana u nedelji tiče potrebno je da znamo u koji je dan pao nulti datum – recimo da je on pao u sredu Tada je dovoljno da odredimo ostatak pn deljenju referentnog broja za 7 (14 mod 7 = 0) i dodamo taj broj na sredu – dobijamo de je 15 septembar pao takode u sredu

Ostaja još da kažemo koji je to nulti trenutak istonje opšte usvojen i kako se izračunava referentni broj ostalih datuma Nulti trenutak koji su astronomi usvojli začudo nije 1 januar 1 godina nego podne 1. **januera 4713. godina pre nove era** – ovakav sistem predlozio ja 1582 godine *Joseph Justus Scaliger* i od tada je u upotrebi Referantni broj datuma računat u odnosu na ovej trenutak se u literetun obično naziva JDN, što ja skraćenica od *Julian Day Number*: mi ćemo koristti termin „referentni broj" ili skraćenicu RB kako bismo izbegli mešanje ovog datuma i julijanskog kelendara o kojem ćemo nešto kasnije govonti

Odluka de se za nulti trenutak usvoji podna a na ponoć je u najmanju ruku neusklađena sa našim uobičajenim poimanjem vremena Pa ipak raziozi za nju su čisto praktične pnrode – većine astronomskih posmatranja se odvija tokom noći, a promena datuma usred akspenmenta unosi konfuziju u dalja računanja Pa ipak, ova konvencija ne predstavlja poseban problem u praktičnim (a pogotovu programerskim) pnmenama pošto u svim proređunima radimo se čitavim danima a ne njihovim delovima Ukoliko međutim pošelite da odredite RB nekog **sekunda** moraćete da uzmate u obzir i ovu astronomsku začkoljicu

Jako smo fiksirali nulti trenutak još na možemo da pređemo na algoritam za računanje referentnog broja Problam je u tome što taj broj neposredno zavisi od korišćenog kalendara, a kalender se kroz istoriju više puta menjao zbog različitih tumačenja pojme „godina" Obično se naime, smatre da Zemlja obide Sunce za 365 dana koliko traja kalendarske godine Ukoliko malo bolje razmislite sehćete se i prestupnih godina – zašto svaka četvrta godina ima dan više? Odgovor znate zato što Zamlja obide Sunce za 365 25 dana što znači de se posle svake četin godine pojavi „greška" od jednog dana koju treba kongovati Kada bismo zaboravili na prestupne godine praznik Nova godine bi se postepeno pomeralo preme jesenjoj ravnodnevici pa bi za 500 godina prvi januer bio jedan od najtoplijih dane ete!

PO STAROM I NOVOM KALENDARU

Astronomi su još u doba starog Egipta uočili ovu anomaliju, pa su uvedane prestupne godine – kalendar kod kojeg je svaka četvrta godina prestupna nazivamo julijanskim, po Juliju Cezaru koji ga je uveo 46 godina pre nove ere (više o tome nešlo kasnije) Ovaj kalendar, međutim, i dalja nije dovoljno tačan – periodi rotacije Zemlje oko sopstvene ose i revolucija oko Sunca nisu međusobno povezani pa bi bilo prilično čudno da je drugi **tačno** 365 25 puta veći od prvog Zamlje se, zasta, ne okreće oko Sunca za 365 25 dana već za nešto manje razlika je doduše mala, ali se iz godine u godinu kumuliše pe se praznik Nove godine pomare prema prolečnoj ravnodnevici što znači de bi pre ili posle prvi januer pao u proleće a docnije i u leto Srednjevekovni astronomi i astrolozi su primetili ovu anomaliju i često na shvatajući njene uzroka (tada se smatralo da Sunce obilazi oko Zemlje!) uvideli kuda ona vodi pa su se pojavili predlozi za revizju kalendara Iako ja nesaglasnost julijanskog kalendara sa prirodom bila nesumnjiva čak i u to vreme nije se bilo lako odlučiti za promenu kalendara, zbog poznate „socijalna inercije" naroda Papa Grgur XIII je „presekao čvor" i proglasio da posle četvrtog nastupa petnaesti oktobar 1582. godine čime je anulirana greška koja se akumulisala za prethodnih 1500 godina. Da bi buduca pokolenja oslobodio sličnih konfliktnih situacija, Grgur XIII je (sledeći savete svog astronoma Luidi Lile (*Luigi Lilio*)) odlučio da godine kojima se završavaju vekovi budu prestupne samo ako je njihov broj deljiv sa 400 tako će 2000. biti prestupna dok su 1900 1800 i 1700 bile proste godine! Ovaj kalendar se naziva gregorijanskim mada treba znati da Grgur XIII nama previše zasluga za njegov nastanak – veći deo posla obavljen je za vreme pontifikata Pija V koji je inicirao reformu kalendara, ali je umro pre nego što je posao završen

Iako je Papin autontet bio nesporan, promena kalendare nije „tak tako" prošla pojavila su se milionska udruženja koje su tražila da im se „vrati njihovih deset dana" a bilo je i relativno ozbiljnih sukoba i nemira Gregorijanski kalendar ja ipak postepeno prihvatan u raznim državame – u britanskim kolonijama (npr današnjim Sjedinjanim Američkim Državama) prihvaćan je 2 septembra 1752 (posle 2 nastupio je 14 septembar) tako da ćete često naći različite podatka o pojedinim istorijskim ličnostima (Džordž Vašington ja roden 11 februara 1732 eli je ovaj datum izražen u julijanskom kalendaru pa se njegov rođendan obeležava 22 februara, negde ćete naći i podatak da je roden 11 februare 1731, pošto je u Britanskim kolonijama tog vremena Nove godina počinjala 25 marta, što je još jedna interesantna anomalija o kojoj treba voditi računa) Neke druge države su prihvatila gregorijanski kalendar znatno kasnije pre svega zbog uticaja Pravoslavne crkve, koja nije prihvatala čak ni matematički zasnovane Papine naredbe U Rusiji je gregorijanski kalendar prihvaćen tak u ovom veku, a ni mi nismo bili bolji – gregorijanski ka-

lendar je u Jugoslaviji (koje se tada zvala Kraljevina Srba, Hrvata i Slovenaca) prihvaćen tek 19. januara 1919. – taj dan je ukazom proglašen za 31. januar. Poslednja moderna država koja je prihvatila gregorijanski kalendar je Turska (1927. godine), mada neki smatraju da je to Rusija, koja se u vreme između dva svetska rata vraćala na julijanski kalendar, da bi najzad ponovo usvojila gregorijanski. Tačan datum preseska na novo računanje vremena u pojedinim državama može da se nađe u knjizi *Explanatory Supplement to the Astronomical Ephemeris and the American Ephemeris and Nautical Almanac* iz 1961. godine, strana 414.

Julijanski kalendar se kod nas održao i do današnjih dana pod imenom „Stari kalendar". Nove godine po ovom kalendaru pada 14. januara. Treba da razumemo da razlika nije u tih četrnaest dana. 2101. godine će se povećati na petnaest, 2201. na šesnaest i tako dalje – za nekih 47.000 godina će se napraviti pun krug pa će se prvi januar julijanskog i gregorijanskog kalendara poklopiti s tim što će se oznake godine razlikovati za jedan!

Bilo koji datum u istoriji sveta ima svoj jedinstveni referentni broj – RB datuma izleska ovih „Računara" je, na primer, 2 449 232. Datum koji pridružujemo tom referentnom broju, međutim, zavisi od kalendara – ako se služimo julijanskim, taj dan će se „zvati" 19. avgust a po gregorijanskom 1. septembar 1993. godine.

Završavajući ovu diskusiju reći ćemo da ni kalendar koji danas koristimo nije naročit. Period revolucije Zemlja oko Sunca se „ne sve ge, danas može izmeniti daleko preciznije, što znači da konstatovana razlika između kalendarske i realne godine postoji i da se slaino akumulira – 4905. godine pojaviće se razlika od jednog dana, tako da je moguće da 4904. godina ne bude prestupna kako bi se ta razlike uklonila. Postoji i predlog po kojem bi godine kojima se završavaju milenijumi bile prestupne samo ako su deljive sa 4000, što bismo mogli da „osetimo" već 2000. godine koja po tom predlogu, ne bi bila prestupna! Osim što je pogrešan, kalendar je krajnje nepraktičan: dužine meseci su različite, prvi u mesecu ne padaju uvek u isti dan, kalendar sa iz godine u godinu pomera za jedan ili dva dana... Promene kalendara su danas u nadležnosti Ujedinjenih nacija koje, verovatno imaju i vidu teškoće sa kojima svet prima novitete, već dvadesetak godina ne mogu da usvoje novi sistem računanje vremena.

UZ MALO MATEMATIKE...

Predimo na algoritam računanja referentnog broja. Za početak ćemo pretpostaviti da svaki četvrti dan po 30. dane i nazvati svaku godinu proisteklu iz te pretpostavke „proširenom godinom" od 367. dana. Pretpostavimo, dalje, da ta proširena godina počinje 1. marta i završava se 30. februara. Ako „pravi" datum obaležimo se DD „pravi" mesec sa MM, a „pravu" godinu se YY i izračunamo y=YY+(MM-3)/12, pokazaće se da se ceo deo broja *y* predstavlja oznaku odgovarajuće „proširene godine" a razlomljen deo broj meseci u njoj. Posmatrajmo sada broj z=367*INT(y) + 367*FRC(y). Prvi sabirak označava 367. dane proširene godine a drugi uzima vrednosti takve da se dodavanjem bilo kog broja *p* iz intervale [7/12, 8/12) dobija (posle odbacivanja decimala) stvarni broj dana protekao od početka godine do početka zadetog meseca. Prema tome, broj INT(367*y+p) daje broj dane koji prethodi zadatom datumu u zadatoj godini naravno uz pretpostavku da su sve godine „proširene" na 367 dana.

Da bismo prešli na prave godine, treba proširenima oduzeti 1 dan ako su prestupne i 2 dana ako su proste. Oduzimanje broja 2* INT(y) se brine o tome: all posle toga treba dodati INT(y/4) da bi se uzele u obzir prestupne godine. Korekcija koju uvodi gregorijanski kalendar zahteva da se od dobijenog broja odbije INT(y/100) a zatim doda INT(y/400). Tako dobijamo broj dane proteklo između DD/MM/YY i 1. marta 0-te godine (nulta godina je ovde čista fikcija pošto je pre 1. godine nove ere bila 1. godina pre nove ere – ipak ova fikcija olakšava računanje, pa je i mnogi astronomi primenjuju) dok za dobijanje „prevog" referentnog broja treba dodati referentni broj pomenutog 1. marta. Njega uzimamo iz tablica i dobijamo formulu:

RB = INT(367*y+p) – 2*INT(y) + INT(y/4) – INT(y/100) + INT(y/400) + DD + 1721119

Formula se može malo „sredriti" izborom pravilnog *p*, kako bi se umanjile numeričke greške i ubrzalo računanje. Ne upuštajući se neposredno u to „sređivanje" koje uz malo pažnje može da izvrši svaki matematički nastrojen čitalac, dajemo konačnu formulu:

RB = INT(INT(INT(367*y) – INT(y) – 0.75* INT(y) + DD) – 0.75*INT(y'/100)) + 1721115

gde je y'=YY+(MM-2.85)/12. Kada se radi sa julijanskim kalendarom, INT(y'/100) treba zameniti brojem 2, što se zasniva na činjenici da se kalendari poklapaju tokom prvih godine nove ere. Funkcija DAT2RB sa slike 1 omogućava izračunavanje referentnog broja za zadati datum.

Listing 1

```
(DD+D+)
zadanom datumu;
{
        Procedura DAT2RB(var---referentni broj
        Copyright (c) 1990-1993 Dejan Aleksandrić
                        "Računari" 94'
}
const gregor: boolean = true; { ovaj parametar se koristi
                                za julijanski kalendar }
{
  Ako je ova funkcija ispravno izračunala referentni broj datuma zadanom od
     strane ulaznih parametara p,m,d
  Ako - dati datum ne postoji (npr. 31. februar)
      ili - nemoguće izračunati referentni broj datuma
      ili - datum pre 1. marta 0-te godine
  Ako izračunati broj ima vrednost -1
}
{
Procedura DAT2RB (dd, mm, yy: INTEGER; { konstanta }
var yy: INTEGER;
BEGIN
  YY:=yy+(mm-2.85)/12.0;
  temp:=INT(367*yy)- INT(yy/4- 0.75*INT(yy/100)+dd;
  IF gregor THEN
    temp:=INT(temp)-0.75*INT(yy/100);
  else
    temp:=INT(temp)-1.0;
  temp:=temp+INT(temp)+721115;
  datbroj:=TRUNC(temp);
END;

{
  DAT      odredivanje dana u broji pod datuma datum
  DAN      dd: =TRUNC datuma, (1-31)
           mm  datuma =TRUNC ((-12)
           yy: datuma godina (1-100000)
  DATUM    nevalja dana (prestupna; (=ponedaljak, ... 6=subota
}
Procedura DAT( dd, mm, yy: INTEGER; var
BEGIN
  day:=(datbroj+dd, mm, yy)+1) mod 7
END;
```

Konverzije datum-referentni broj

Listing 2

```
{
    datuma - konverzija referentnog broja dana u datum
    ULAZ     -dat: referentni broj dana, datumuk od
               1. januara 4713. godine p n e
    IZLAZ    dd: =nevalja datuma (1-31)
             mm: =nevalja datuma (1-12)
             yy: =nevalja godina (1-100000)
}
Procedura rb2dat (rbr: zadanom; var dd mm,yy: INTEGER);
var d : real;
    dy,my,ddy,mdy: INTEGER;
BEGIN
  d:=rb-(72(((
  d:=TRUNC(((a-0.2(/36524.25);
  IF gregor THEN
    dy:=a-TRUNC(r/4(
  ELSE
    dy:=a;
  yy :=TRUNC((dy-0.2)/365.25(;
  mdy :=a-TRUNC((365.25*yy);
  mm :=TRUNC((dy-0.6)/30.6);
  dd :=TRUNC(dy-30.6*mm+0.5);
  IF mm>9 THEN BEGIN mm:=mm-9; YY:=YY+1 END
          ELSE mm:=mm+3;
END;
```

Konverzije referentnog broja dane u datum

Inverzna operacija je za nijansu složenija. Najpre ćemo od zadatog RB oduzeti 1721119 i tako dobiti N, broj dana proteklih od 1. marta 0-te godine do zadatog dana. Sredinje dužine gregorijanskog veka je 36524.25 dana, dok je broj gregorijanskih vekova protekao između tekućeg datuma i 1. marta 0-te godine C=INT((N-e)/36524.25) gde je e bilo koji broj iz intervala (0, 0.25], mi ćemo uzeti e=0.2. Broj N'=N + C – INT(C/4) daje broj proteklih dana računat po gregorijanskom kalendaru dok N'=N+2 odgovara julijanskom. Delje računanje je isto za oba kalendara uz uzimanje ispravnog N.

Neka su Y' i M „pomerene" godina i mesec računati kao da godina počinje 1. marta (M=0). Tada je Y'=INT((N'-e')/365.25) gde je e' iz istog intervala kao i e (uzećemo e'=e=0.2). Broj dana protekao od početka „pomerene" godine je N' = N' – INT(365.25* Y'), dok je M' = INT((N'-d)/30.6) gde je *d* bilo koji broj iz intervala (0.4, 0.6) mi ćemo uzeti *d=0.5*. Pošto smo izračunali ova brojeve ostaje samo da se vratimo na „normalnu" godinu: biće DD = INT(N' – 30.6*M + 0.5) i za M>9 YY=Y'+1 i MM=M-9 a za M<=9 YY=Y' i M=M+3. Procedura RB2DAT sa slike 2 omogućava izračunavanje datuma, meseca i godine za zadati referentni broj.

U STARIM DANIMA

Programe se slika 1 i 2 možete „bezbedno" primenjivati na sve datume iz bliže prošlosti i budućnosti naravno sve do dana kada dođe do reforme kalendara. Kada se radi o datumima iz dalje prošlosti, a naročito o prvim godinama posle Hrista treba biti veoma oprezan. Listajući neke stare knjige, primetićete nesaglasnosti između datuma dana u nedelji i formule koje smo izneli. Stvar je u tome što, čitajući neku steru knjigu obično na čitate originali rukopis nego ko-zna-koji-po-redu prepis i prevod – uostalom malo je verovatno da bista se uopšte snašli sa starim pismima i jezikom. Svaki prevod unosi poneku nepreciznost, dešave se da se pri tome neki datum konvertuje na „novi sistem" ili se ne konvertuje, pa se posle desi da pri sledećem prevodu već konvertovan datum bude greškom još jednom konvertovan – za 2000. godina te greške su se „akumulisale" tako da pri tumačenju datuma iz antičke istorije treba biti veoma oprezan. Zanimljiv osvrt na tu temu objavio je 1972. godine *A E Samuel* pod naslovom „Calendars and Years in Classical Antiquity" a vredi pročitati i tekst „Calendars – the Real Story" koji je *William E Hitchins* objavio marta 1982. u *PPC Journal*-u v9n2p36. Ovde ćemo se samo ukratko osvrnuti na nekoliko najzanimljivijih činjenica.

Bilo kakvo ostanjanje na julijanski kalendar pre 46. godine p.n.e. nema mnogo smisla – Rimljani su i tada koristili sličan kalendar (za pravo varijaciju egipatskog kalendara) ali je dodavanje „prestupnih" dana (zapravo prestupnog meseca *intercalary*) vršeno stihijski i neprecizno. Što zbog neznanja, što zbog nepažnje, što iz političkih razloga (dodate godine još jedan mesec i odmah su porezi veći). Tih „političkih" promena kalendara se sakupilo toliko da je 46. godina p.n.e. po svemu sudeći imala januar (29 dana), februar (24 dana), interkalar (27), mart (31), april (29), maj (31), jun (29), quatilis (31), sextilis (29), septembar (29), oktobar (31), novembar (29), undecembar (34) i decembar (29 dana).

Boraveći u Egiptu i „družeći" se sa Kleopatrom Cezar je 48. godine p.n.e. upoznao astronoma *Sosigenesa* koji mu je predložio us-

vajanje fiksnog kalendara zasnovanog na egipatskom računanju vremena koje uključuje prestupne godine. Cezaru se ideja okončanja „kalendarskog haosa" dopala, pa je *Sosigenes* u saradnji sa *Marcusom Flaviusom* prilagodio ovaj kalendar rimskim običajima ukidajući „prestupni mesec" i raspoređujući njegove dane ostalim mesecima. Kalendar je predočen Rimskom Senatu 46 godine p n e i, po usvajanju, postao je zakon. Naravno ova godina sa po tom zakonu ne naziva 46-tom godinom pre nove ere (svakom je pokoljenju njegova ale „nova") već je „nulti" datum vezan za nestanak Rima i u prilično man proizvoljno fiksiran na 753 godinu p n e. tako de se kaže da je julijanski kalendar stupio na snagu 1 januara 709 godine A.U.C.

Godinu dana kasnije Senat je predložio da se mesec *Quantilis* preimenuje u *Iuhus* (Jul), u slavu Julija Cezara (zbog složene birokratske procedure ovaj predlog nije postao zakon sve do posle atentata na Cezara 15 marta). Međutim Cezarov kalendar u početku nije primenjivan na način koji je *Sosigenes* zamislio tako da je svaka treća godina bila prestupna – Senat je tada bio previše zauzet raznim političkim previranjima da bi sa ozbiljnije bavio „nametanjem" kalendara lokalnim vlastima. Pošto je Oktavijan August čvrsto zaseo na presto, primećena je nesaglasnost kalendara se realnošću, pa je odlučeno da između 8 godine p n e i 8 godine n e ne bude prestupnih godina tako bi se stvari dovele u red. Dakle, 4 godina n e nije bila prestupna, pa naše programe sa slika 1 i 2 ne treba primenjivati na datume pre marta 4 godina – mogli smo naravno da unesemo i ovu korekciju, ali smo smatrali da bi ona samo nepotrebno komplikovala proračune. U sto vreme Senat je predložio da septembar bude preimenovan u avgust u slavu imperatora koji je rođen 23 septembra 63 godina p n e (dobro je poznat Klaudijev komentar ne ovu temu „šta ćemo raditi kada se rodi Innaash Cezar?"), ali ja sam August iz razloga oko kojih se isključan ne slažu, izabrao mesec *Sextilis*. Pominjanje Oktavijana Augusta je ujedno dobra prilika da ukažemo na jednu od zabluda kojom su opterećena naša znanja o kalendaru – verovatno ste u školi učili da je August, da „njegov" mesec na bi bio kraći od „Cezarovog" jula, oduzeo jedan dan februaru i dodeo ga avgustu. Teorija zvuči logično (zašto bi inače februar imao manja dana od bilo kog drugog meseca?) ali je istorijska istraživanja ne potvrđuju – dužina svih meseci su određene još u doba Julija Cezara a februar ima malo dana zato što je smatran „baksuznim mesecom" (posvećen je nekim nižim bogovima), pa ja bilo pogodno da bude što kraći. Teorija o produžanju avgusta pojavila se u XIV veku i niko ja nije uzimao za ozbiljno dok se krajem prošlog veka nije pojavila u jednom od tomova *Encyclopedia Britannica* odakle je pranesena u mnoge udžbenike. Redakcija ova slavna enciklopedija je početkom osamdesetih izjavila de je ova greška otklonjena u najnovijem izdanju, ali nismo imali prilike da provenmo da li je to izdanje i objavljeno.

Danas važeće imenovanje godina predloženo je 532 godine kalu đer *Dionysis Exiguus* (poznat i pod imenom *Denys le Petit*) – godina Hristovog rođenja je nazvana 1 godinom A.D. e godine pre nje 1 godina B.C. tako *Exiguus* u svojim proračunima nije bio baš precizan, njegovo računanje vremena je široko usvojeno čak i u nehrišćanskim zemljama. U nekim zamljama je kasnije pokušane revizija „prve godine" (u doba Francuske revolucije usvojen je novi kalendar a pokušaje je činio i Staljin sa „Sovjetskim kalandarom"), ali ni jedna od tih

Listing 3

```
PROGRAM si;
{
                    dr_domenovic
  copyright (C) 1987-1993 Dejan Ristanović
                  Nebojšin 98
}
USES dos, crt;
FUNCTION st (ulaz: LONGINT): string;
BEGIN str(ulaz,st);
END;
FUNCTION dan(g,m,d:integer):integer;
VAR rez: integer;
BEGIN
  IF d<13 THEN rez:=d-1+(31*(m-1)) ELSE rez:=-97+(d);
  IF m<3 THEN rez:=rez+g-1 ELSE rez:=rez+g / 4*(3+1);
  dan:=g-rez+    +rez;g
END;
VAR d,m,g: integer;
    napoc, sad, ldn, ldn, ldn, kdan, razl: longint;
    dn, mn, gn, no, MESEC;
    odg: string;
BEGIN
  writeln;
  write(' Unesi datum rođenja (DD   ');
  readln (d);
  write(' Mesec              (MM   ');
  writeln (m);
  write(' Godinu             (YYYY   ');
  readln (g);
  REPEAT
    write ( #(napomenuvode) !!! #(lodani);  !);
    writeln (odg);
    odg:=UPCASE (odg[1]);
  UNTIL (odg='N') OR (odg='D');
  gotovo:=(odg  no  da  ne);
  IF (odg='D') THEN
  BEGIN
    napoc:=dan (1,1,1);
    sad:=dan(gn,mn,dn);
    ldn:=sad-napoc;
    kdan:=ldn MOD 7;
    IF kdan=0 THEN
    BEGIN
      kdan:=7;
      razl:=(gn-1)*365+(gn-1)DIV 4+(gn-1)DIV 100+(gn-1)DIV 400+ldn;
      writeln ('Od početka računanja vremena je prošlo ',razl,' dana');
    END
    ELSE
    BEGIN
      kdan:=kdan DIV 7;
      writeln ('Od početka računanja vremena je prošlo ',kdan,' nedelja');
    END;
  END;
END.
```

SI rođendani

alternativa nije se duže zadržala na istorijskoj pozornici.

SI RODENDANI

Da se ovaj tekst ne bi završio potprogramima dajemo i praktičan primer njihove primene, koji će vam omogućiti da zabavite i iznenadite prijatelja. Svi oni osim eventualno stanovnika „Alise u zemlji čuda" slave rođendan svaga jednom godišnje. Ali zašto baš jednom godišnje? Godine i nije neka jedinica za merenje vremena videli smo da se Zemlja ako ništa drugo ne okrene oko Sunca za 365 pa čak ni za 365 25 dana! Sekund je se druge strane precizno definisana jedinica za merenje vremena koja je kao takva ušla i u SI sistem. Zašto se rođendani ne bi slavio kade se na primer, navrši jedna gigasekunda?

Program sa slike 3 zahteva da otkucate dan mesec i godinu rođenja zatim preuzima tekuće vreme očitavajući PC-jev časovnik i pronalazi datum vašeg sledećeg „SI rođendana" kao i starost koju ćete toga dana navršiti. Samo se po sebi razume da računar koji niste snabdeli podatkom o tačnom **vremenu** vašeg rođenja na može da kaze *kog ćete sekunda* u toku naskog dana navršil određen broj megasekundi ali je i datum obično sasvim dovoljan. Vremena se izražavaju u megasekundama a „rođendan" nastupa kada napunite punu sloiu megasekundi – ovu smo brojku izabrali jer je gigasekund prevelika jedinica a vi svakako ne želite da rođendan slavite samo jednom u 32 godine!

Kad o bi vam rođendani bili češći program smo osposobili da izračunava i vašu starost u kilodanima – izabente jedinicu koja vam više odgovara i iznenadite nakog prijatelja koji zasigurno nije znao da mu je baš danas rođendan!

STA IMA NOVO

SOFTVER

Alatke

Brža štampa iz Windows-a: SuperPrint 3.0

Iz firme „Zenographics" stiže *SuperPrint 3.0* dopradena verzija paketa za ubrzanje štampanje iz *Windows*-a, e uz nju i *ZScript* 32-bitni PostScript interpreter za *Windows*. Novi *ZScript* je, po rečima proizvođača u stanju da prevodi PostScript datoteke sa bilo koje platforme u *Windows 3.1* format.

SuperPrint 3.0 se dobija u tri verzije – za desktop za profesionalne namene i za mreže i po prvi put sadrži *SuperRIP* (*Raster Image Processor*) 32-bitni softverski RIP koji deje performanse komercijalnog RIP a.

Zahvaljujući direktnoj kontroli procesora, ovaj novi RIP postiže četiri puta bolje performanse u odnosu na 8-bitne *Windows 3.1* drajvere. Još jedna novost je *SuperFax*, alatka za upravljanje CAS-kompatibilnim fax karticama.

Tu su i dalje funkcije poznate iz prethodnih verzija – *SuperQueue*, predviđen da zameni sporiji *Windows Print Manager* kao i *Supertask* alatka za skaliranje fontove koja omogućava trenutno skaliranje slovnih likova iz četiri osnovna izvora – Adobe Bitstream Intelifont i NimbusQ bilo na ekranu ili na štampaču.

Desktop verzije programe *SuperPrint 3.0* koje podržava većinu matričnih *ink jet* i laserskih štampača, ima preporučenu cenu od 149.95 USD e u paketu sa *ZScript*-om 295 USD.

Kao potencijalna primene ZScripta pominju se pre svega konverzija EPS (*Encapsulated PostScript*) datoteka specijalnih *PostScript* alatata (na primer tekstura za popunjavanja objekata u paketu *CorelDraw*) i *Adobe Type 3* slovnih likova u *Windows* format. Nakon pravođenja, datoteke se mogu pregledati i kopirati u *Windows Clipboard* odakle se uvoze u druge *Windows* aplikacije ili odštampati putem *Super Print* 32-bitnih li drugih *Windows* drajvera za štampače.

Konsne adresa: Zenographics, tel. 714-851-6352

Softverske alatke

Altia Design 1.2: Dinamički interfejs

„Altia" softverska kompanija iz Kolorada, najavila je verziju 1.2 svog proizvoda *Altia Design*. Kako napominja proizvođač, to nije alatka za izgradnju grafičkog korisničkog interfejsa (GUI) nili paket grafičkih alatki koji konsti ograničen skup komponenti standardnog korisničkog interfejsa. *Altia Design* je nešto drugo – omogućava dizajneima da kreiraju dinamički interfejs potpuno prilagođen korisniku koji precizno simulira fizička funkcije i prikazuje podatke i informacije u stvarnom vremenu.

Altia Odesign se prvi put pojavio u avgustu 1992 i koristi se za dizajn raznih prozrovela i sistema od simulatora mrežnih telekomunikacija preko monitora kritičnih poslova pa sva sile osciloskopa, ultrazvučnih aparata i komunikacione opreme. Pri usavršavanju alatki i ukupnom kraianju verzije 1.2, korišćena su iskustva i predlozi ranijih korisnika.

Program sadrži grafički editor koji omogućava lako spajanje već izgrađenih grafičkih komponenti kao što su dugmad merači brojčanici CRT displej tastelol tasteri. Služi i za kraianje novih komponenti čime se eliminiše potreba da specijalisti za softver pišu hiljada linija grafičkog koda de bi postigli isti rezultat.

Niz interaktivnih opcije za animaciju olakšava kreiranje dinamičkih komponenti a tu je i puna kontrola nad inputom interaktivna definicija input prostora i radnji tako da korisnik barata komponentama pomeću miša dodirom ekrana III drugim input uređajem. Opcija za snimanje i plejbek beleži interakcije korisnika prikupija podatke o ljudskom faktoru tako da se može koristiti za *on-line* obuku i automatski demo. *Altia* nudi i verziju za ogradnju u vec gotove proizvode.

Verzija 1.2 podrževa uvoz kolor bit-mapa iz drugih sistema za dizajn ili sa skenera kao i Tadpole SparcBook prenosivu SPARC radnu stanicu. *Altia Design* se može integrisati u sve *Motif* i druge *X Windows* aplikacija.

Celokupan *Altia Design* sistem košta 7 900 dolara.

Konsne adresa: Altia – 719-598-4299, fax 719-598-4392)

INSTITUT ZA NUKLEARNE NAUKE "VINČA"
CENTAR ZA PERMANENTNO OBRAZOVANJE

11000 Beograd, Nemanjina 4/X
Telefon: 011 / 683-390, 682-486, 641-155 / 107, 181
Telefax: 011 / 682-486

KURSEVI PROGRAMIRANJA I PRIMENE RAČUNARA

AutoCAD
AutoLISP
AutoSHADE
3D STUDIO
SOLID
CorelDRAW!
AEC ARCHITECTURAL

WORDPERFECT
MS WORD
WORDSTAR
CHIWRITER
MULTIMATE
AMI PRO
WS 2000+
WORD FOR WINDOWS
WORDPERFECT FOR WINDOWS
VENTURA PUBLISHER

ADA
PL / 1
MODULA 2
PROLOG

ORACLE
CLARION
CLIPPER
FoxBASE
MAGIC
dBASE
FoxPro

MathCAD
REDUCE
EUREKA
SURFER
GRAPHER

VMS
UNIX
XENIX
OS / 2
MS-DOS
NOVELL
WINDOWS

TURBO C
C++
COBOL
PASCAL
FORTRAN 77

QUATTRO PRO
MS WORKS
MS EXCEL
FRAMEWORK
PARADOX
LOTUS 1-2-3
AGENDA
Freelance Graphics
ec:Mail
SYMPHONY
SPSS / PC+

PRIMAVERA
SUPERPROJECT

MACROASSEMBLER

» Jedini autorizovani obrazovni centar u Srbiji softverskih firmi **"AUTODESK LTD", THE SANTA CRUZ OPERATION", "BORLAND" i "LOTUS".**
» Nastava se održava u Beogradu, Nemanjina 4/X. Centar poseduje preko 70 računara.
» Broj polaznika je ograničen (dvadeset u grupi, do dva slušaoca za jednim računarom).
» Posebne pogodnosti za preduzeća (kursevi kod korisnika, kursevi vaših programa, popust za više kandidata).
» Slušaoci dobijaju literaturu i, po uspešno položenom završnom ispitu, uverenje o završenom kursu.
» Mogućnost plaćanja kotizacije za kurs u ratama.

TSR UZVRAĆA UDARAC

Jedna od intaresnatnijih posledica taksta o rezidentnim programima iz prošlog broja je količina pitanja koja su se iz njega izrodila. Naime, mada se autor trudio da pomene sve što bi moglo biti potrebno za pisanje jadnog pravog rezidentnog programa, čitaoci su (u aprazi sa mračnim silama haosa i bezumlja) pronašli još nekoliko intarasantnih problema koji nisu bili rešeni. Recimo, može li Borland C++ da proizveda rezidentni program čija će valičina biti manja od 100 bajtova? Odgovor je, pogadate, da moža...

Bojan Petrović

PROBLEM PRVI: NAJNAJNI NA SVETU

Pre nego što se upustimo u suvoparne teorijske rasprava, malo ćemo otvoriti apelite Evo dakle šta nas čeka pokušaćemo da napišemo „rezidentni" program čija će valična biti ispod stobnjek bajtova, koristeći Borland C++ Ako se malo prisetite u prošlom broju smo pokušali nešto slično (da što više smanjimo veličinu rezidentnog dela) ali se nismo baš proslavili i to iz više razloga pre svega prošli put smo insistirali na „pamet" programa fleksibilnosti teorijskim objašnjenje a samo smo dolakli način kojim bi se veli čina programa mogla smanjiti drugo ono što ćemo sada raditi neće baš biti u pravom smislu reči rezidentni program jer za razliku od svih lakvih, neće ostati aktiven pozivanjem „Terminate and Stay Resident" (TSR) funkcije DOS prakide Konačno ovde ćemo se služiti nekim osobenostima Borlandovog prevodioca. tako da prebacivanja ne neki drugi kompajler može biti problematično

Prisetimo se još malo prethodnog nastavka rekli smo da je korišćenja mamorijskih modela i segmenala sjajan način da se po pozivanju gora pomenute funkcije DOS-a otarasimo nepotrebnog dele kode Tada smo nepotrebni deo programa (sa gledišta rezidentnog rada) postavili u poseban segment (segmente) i sačuvali samo ostatak, čima smo odbacili sve nepotrebna rutina koja su se recimo bavile ispisom lepih poruka i pomoći pri Installranju čitanjem opcija sa komandne linije deinstaliranjem već rezidentnog programa i slično. Problam koji se tu javio je nezaobilazna činjenica de u svaki program koji radite moćnim prevodiocem kao sto je pomenuti ulaza isto lako moćne biblioteke koje su u nekoj ełapi rada neophodne Medutim rezidentni program obično ima jako usko polje delovanja, kome nisu potrebne kojekakva biblioteka već samo nekoliko osnovnih funkcija Dakle prethodni nastavak nam je dao ideju kako da napravimo „inteligentan" rezidentni program a ovaj put ćemo pokušati pristup sa druge strana napravićemo jedan jako glup ali maksimalno mali TSR izbacujući potpuno sve biblioteke koje nam nagura kompajler

IDEJA ..

Ideja se, ukratko svodi na sledeće ako „Terminate and Stay Resident" funkcija ostavlja u memoriji zadati broj paragrafa od lačke gde počinja program postoji li nečin da iz celokupnog programa lščupamo samo deo koji nam zaista treba (recimo iz sredine) I samo njega sačuvemo, a ostalo obrišemo? Sad bi neko rekao „pa nismo li baš to učinili prošli put smeštajući ne sam kraj programa nepotrebne delove pa njih odsecajući?" što i nija dalako od istine Nažalost tima smo od sekli samo naš „lični" deo programe e ne i biblioteke koje po definicji dolaza na početak (vidi ilustracije raspored segmanala u *Large* modelu u prošlom broju) Njih ovom matodom na mozemo normalno odbaciti Ima li bolja ideja? Naravno

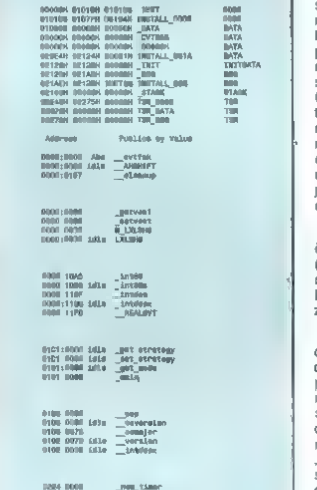
Ideje proizlazi iz načina na koji DOS odvaja prostor u memoriji za programe o čemu je već više puta bilo reči (recimo Računeri 77, tekst „DOS 5 0 ispod haube") ali nije zgoreg rekapitulirati Memonja je u DOS u organizovane u obliku lanca blokova gda svaki blok poseduje svoja za glavlje („Memory Control Block", MCB) MCB sadrži važna podatke o bloku memonje koji sledi kao i: o položaju bloka unutar lanca blokova Ne ofsatu 00h zaglavlja je jednobajtna oznaka o položaju u lancu gde poslednji od njih dobija oznaku „Z" a svi ostali „M" Ofsal 01h sadrži dvobajtnu segmentnu adresu vlasnika datog bloka što zapravo pokazuje ko je taj blok razarvlsao za sebe Recimo ako je „vlasnik" 0000h reč je o nedodeljanom bloku (slobod- u mamonja) ili ako je vlasnik 0008h blok pripade sistemu Svi izvršni programi za vlasnika navode samog sebe (odnosno adresu svog početka), dok okoline programa („anvironment") kao vlasnika uzima program kome pripada istovremeno ovaj podalak kazuje DOS-u koje blokove treba da smatra kao jednu celinu odnosno, ukoliko neki program prestaje se radom DOS na brže samo njega iz memonje, već i sva blokove čiji je on vlesnik Očito je uz grad da ovda na govorimo o fizičkom brisanju, nego o prostoj promeni vlasnika na 0000h Konačno, na ofsetu 0003h MCB čuva i veličinu bloka koji sledi tako da se recimo početak sledeceg zaglavlja moza lako izračunati kao adresa datog MCB + dužina bloka koji sledi + 16 bajtova za dužinu samog zaglavlja I samo da pomenemo MCB u verzijame DOS-a počevši od 4 0 čuva i ima programe o kome je rec sto moza biti korisien podalak

Šta se, dakle dešava kada „startujemo" neki program? Pošto je sistem odgovoran za njega, pronalazi sa kontinuelan blok memonje u koji može da se smesti progaram uvećan za 16 bajtova potrebnih za MCB U zaglavlje se upisuju odgovarajuće oznake i podaci (vlasnik se postavlja na segment gde počinje program) I krece se sa radom Zalim recimo iz programa se pozivom od govarajuće funkcije DOS a odvaja još jedan blok memdrija za neke podatke I on automatski dobija svoje zeglavlja (što čini pomenuta funkcije) Neravno vlasnik sada neće biti adresa gde počinja lej novi blok, nego će to biti segmentna adresa programa iz koga je pozvana DOS funkcija za njegovo stvaranje Konačno, kada program završi sa radom DOS u njegovo zaglavlje upisuje 0000h za vlasnika, i protrči kroz lanac blokova tražeći blokove čij je on vlasnik bio Tu nailazi ne onaj blok podetaka pa „brisa" i njega. Konačno da pomenamo i to pri svakom „brisanju" DOS osmotri susedna blokova mamonje i ukoliko je neki od njih takoda slobodan (0000h vlasnik) spaje ge sa ovim susednim Na taj način se obezbeđuja što manja fragmentacija memorije

Program koji želi da postane rezidentan, to čini na jedan vrlo trivijalan način posebnom (TSR) funkcijom se DOS obaveštava da taj-i-taj deo memonje treba de ostane I dalje u lancu blokova kao aktivan program lako je on zvaničnо završio posao tako da DOS ne dira MCB

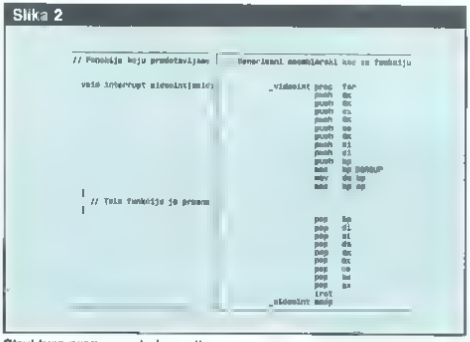
I konačno evo ideja kako bi bilo da mi u okviru instalecije rezidentnog programa, „peške" otvorimo jedan dovoljno veliki blok memonja popunimo odgovarajući mu MCB (namemo promenimo oznaku o vlasniku kako ga DOS ne bi obrisao) u blok smestimo samo rutine koje treba de ostanu aktivna i njima potrebne podatke uključimo taj blok u lanac DOS blokova i ne pozivajući „Terminate and Stay Resident" funkciju završimo sa radom, bnsuci tako seba iz memonje? Posledica će biti sledeća startovanjem našeg programe će se prvo otvoriti jedan blok memorije gda će se odvijati rad naša instalecija zatim na na kom drugom meslu (za sade neodređenom) još jedan blok gde će se pramestiti deo rutina, a potom će DOS onaj prvi blok proglasiti slobodnim, a ovaj drugi neće ni taknuti Nije loše

Slika 1



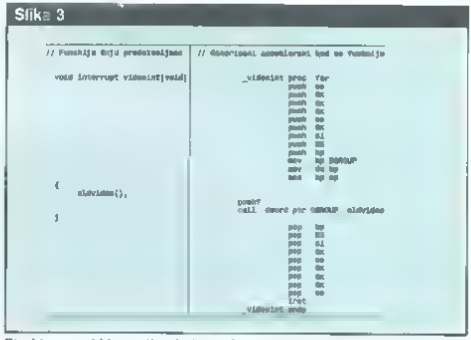
Struktura prevedenog programa uz raspored segmenata

Slika 2



Struktura prazne prekidne rutine

Slika 3



Struktura prekidne rutine koja poziva neku drugu

...I ZAČKOLJICE

Iznasena idaje sasvim lepo fukcioniše na papiru, ali će u praksi zahtevati malo prepravki. Najveći problem uopšte nije u idaji nego u načinu na koji DOS reda memorijske blokove u pomenutom lancu. Pri odvajanju novog bloka memorije podrazumevani način prelrage lanca blokova za slobodnim prostorom je „bottom-up", odnosno od prvog (najnižeg) lanca u bloku ka krajnjem (najvišem). Proces instalacije rezidentnog programa bi zapravo lakao ne sledeci način. DOS prvo odvaja blok memorija za program iz koga camo instaliral rezidentni program i u njegov MCB upisuje oznaku vlesnika, iz tako pokrenutog programa mi sami od DOS-a zahtevamo blok memorije popunjavamo podatke u njegovom MCB manjejuci vlasnika. selimo rutine koje su nam potrebne i završavamo sa radom, završiak rada je znak DOS-u da trenutno aktivni blok nije više potreban i on ga na već pomenuti načini proglasava slobodnim a pošto nije „nadležan" ze ovaj rezidentni blok njaga ostavlja u lancu. Posledica ovog je da u lancu blokova ostaje „rupe" slobodnog prostore veličina programa iz koga smo vršili instalaciju. Ovo znači da pri aktiviranju sledeceg programa pri ilaženju slobodnog prostora u lancu za njega DOS prvo nailazi ne ovo parče slobodnog prostora koje verovatno nije dovoljno veliko, prelazi na rezidentni program i posle njega konačno nailazi na dovoljno veliki slobodan prostor. Dakla, u velikom procentu slučajeva, pomenuta „rupa" će za DOS biti neupotrebljiva (naravno ovde smo potpuno zanemarili postojanja bloka podataka o okolini programa za koga se lakođe odvaja poseban memorijski blok. mada je njegova veličina reda veličina nekoliko slotina bajtova. Iako da može da stana u pomenutu „rupu" i dalje će veći deo log prostora ostali neiskorišćen) Ova fragmentiacija memorije čije opasnosti raste sa veličinom „rupe" nija pozeljna pa mora mo smisliti način da je izbegnamo

Rešanja smo već nagovestili. Baš kao što je podrazumeveni nacin kralanja po lancu blokova „bottom up" lako možemo u nekom intlervalu vremene DOS u neredni upolrebu nekog drugog. U našem slučaju bio bi najzgodniji „top-down" jer bi tako rezidentni blok ostao na samom vrhu lanca blokova (pošto se mesto trezi odozgo na dole logično ja da ćemo prvo naleteli na onaj najveći prazan blok memorije pa će DOS shodno načinu pretraživanja sa njegovog kraja „otkinuti" parče potrebne veličine). Kasnije bi se ona naša rupa stopile se sledećim blokom koji je po pravilu slobodan prostor (jer smo do log trenutka konstili „bottom-up"). Tima smo obezbedili postojanje najvećeg moguceg komada slobodna memorija čima je problem rešen. Mada sa druge strane pošlo je ovakav nacin remecenja ustaljenog načine rada sa lancem blokova neobican može se desiti da naki programi imaju problema sa neslobodnim blokom na samom kraju lanca

Na kraju posebna napomena. radi obezbedi venja ispravnog funkcionisanja DOS-a bilno je po smeštanju rezidentnog dela na svoja mesto vratiti sistemu stari način rada sa lancem blokova (naravno niko ne garantuje da je DOS uvak u ovom ili onom načinu rada tako da se mod rada mora negda zapamtiti kako bi znali u koji mod da vratimo DOS po završetku našeg žongliranja)

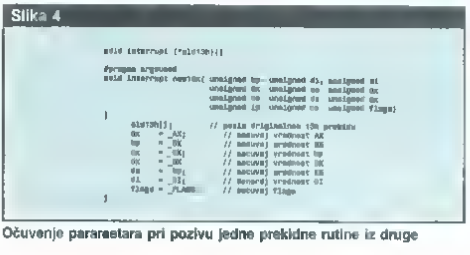
TEHNIČKE STVARI

Sade nam ostaje da rešimo par tehničkih problema. Naime baš kao i prošli put, bice nam potrebno da tačno znamo gde su rutina od interesa (i odgovarajuće promenljiva). de bismo mogli da ih presalimo u novootvoreni blok memorije. Upotrebićemo štos iz prošlog nastavka smeštajući sve rutine i vanjeble rezidentnog dela u poseban modul za koga cemo zatim proglasiti nove CODE DATA i BSS segmente. e koji će biti postavljeni ne sam kraj ukupnog koda. Na taj način ćemo izdvojiti celinu koja nam treba. ali i dalje ne znamo gde ona tačno počinje (po adresi) a gde se zavrsave. Ne sreću novi CODE segment se smešla prvi od pomenutih segmenata tako de će početak potrebnog dela biti zapravo adresa funkcija kojom preuzimamo tajmer prekid (vidi listinge). Kraj ćemo saznati uvođenjem dodatne promenljive koje će biti neiskorišćena u modulu. pa će zbog svoje prirode biti smeštena u BSS segmenl koji se od pomenuta tri segmanta smešta poslednji. Duzinu koda koji premeštamo računamo kao prostu razliku između ofseta prekidne rutine (**new_timer**) i pomenute promenljive (**tsr_and**)

Kao ilustraciju ovoga, pogledajmo sliku jedan gde je izveden raspored koda po segmentima za pravedan program sa listinga. Da bi ilustracija bila još efektnija preimenovali smo CODE DATA i BSS segmenle instalacionog modula, tako da je lakše videti šta smo postigli pomeranjem segmenata TSR modula

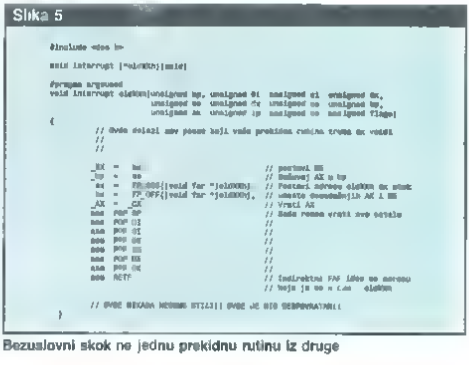
Dalje činjenica je da u rezidentnom dalu ne možemo koristit klasične pozive funkcijama i promenljivima jer zapravo pišemo relokabilni kod

Slika 4



Očuvenje pararaetara pri pozivu jedne prekidne rutine iz druge

Slika 5



Bezuslovni skok ne jednu prekidnu rutinu iz druge

tsr.c

```
#pragma option -zCTSR_CODE      /* CODE segment */
#pragma option -zPTSR_GROUP     /* CODE group   */
#pragma option -zATSR           /* CODE video   */

#pragma option -zRTSR_DATA      /* DATA segment */
#pragma option -zETSR_GROUP     /* DATA group   */
#pragma option -zTTSR           /* DATA class   */

#pragma option -zBTSR_BSS       /* BSS segment  */
#pragma option -zGTSR_GROUP     /* BSS group    */
#pragma option -zDTSR           /* BSS class    */

#pragma option -k               /* Use standard BSOBC Trans  */
#pragma option -O               /* Start checking OFF  */

#pragma option -Os              /* Generate the smallest code possible */

#include <dos.h>
#include <pragma.h>


void interrupt (*old_timer)(void) = (void interrupt (*)(void))NULL;
unsigned short _psp *screen_seg = (unsigned short *)0x0B8A;

#define SCREEN_SEG (*(short _psp * _cs *)(FP_OFF(screen_seg)))
#define OLD_TIMER (*(void interrupt (* _cs *)(void))(FP_OFF(old_timer)))

void interrupt new_timer(void)
{

        SCREEN_SEG[79] = (0x0500 + (*(unsigned short far *) MK_FP(0x40  0x006C) & 0xFF));

        OLD_TIMER();
}

char tsr_psp,
```

Telo minimalnog rezidentnog programa

Listing 3

```
#include <dos.h>
#include <malloc.h>

void interrupt (far *oldvideo)(void);

#pragma argsused
void interrupt videohint( unsigned _bp  unsigned di, unsigned si
                          unsigned ds  unsigned es  unsigned dx,
                          unsigned cx, unsigned bx, unsigned ax,
                          unsigned ip  unsigned cs  unsigned flags)
{
  asm push bp
  asm mov bp, sp

  oldvideo();

  asm pop bp
  asm[_AX];
  asm[_BX];
  asm[_CX];
  asm[_DX];
  asm[_SI];
  asm[_DI];
  asm[_BP];
  die_Bl;
  al= Bl;
}

void main()
{
  oldvideo=getvect(0x10);
  setvect(0x10, videohint);
  keep(0, (_SS+(_SP/16) -_psp));
}
```

Rešenja problema se VIDEO prekidom

Instalacije reinimelnog rezidentnog progreme

install.c

```
#pragma option -zCINSTALL_CODE          /* Inar ime za CODE segment */
#pragma option -zDINSTALL_DATA          /* Inar ime za DATA segment */
#pragma option -zBINSTALL_BSS           /  nevo ime za BSS segment  */

#include <dos.h>
#include <pragma.h>
#include <malloc.h>
#include <stdio.h>

extern Inar Tsr_End;
Inradon unsigned short _psp *far screen_seg;
extern void interrupt (Tar new_timer)(void);
extern void interrupt (*far old_timer)(void);

CHAIN CODE regs

unsigned short far strategy(Inar)
{
        regs.x.ax = 0x0000;
        Inradon(&regs  &regs);
        return regs.x.ax;
}

void far strategy(unsigned char Str0r)
{
        regs.x.ax = 0x0000
        regs.h.bl = str0r;
        regs.h.dx = 0
        Inradon(&regs  &regs);
}

unsigned char get_mode(void)
{
        regs.h.ah = 0x0F;
        Int10h(0x10  &regs  &regs);
        return regs.h.al;
}


Int main(void)
{
        char  key *dos_psp
        unsigned Inert old_strategy;
        unsigned short Tsr_Size
        unsigned short parse;

        if (_argc!=2 || 0)
        {
                fprintf(stderr  "Use as: program.exe parse HH DOS 3.0 Varsi!=    "||
                return (1);
        }

        Tsr_Inar = FP_OFF(&Tsr_End)   FP_OFF(dos_Timer);
        parse = (FP_OFF(*str_arg) - 10) / 16;
        if (parse < 4) parse = 4;
        if ||(get_mode() & 0x7F) == 7)
                screen_seg = ||unsigned short _psp *)0x0000);
        else
                screen_seg = ||unsigned short _psp *)0x0000);
        old_timer = getvect(0x1c);

        /* zapamtvjam staro strategija rada za Inmeam blokova i postavlja
           Inar  CODE je Inadno se top down                                */
        old_strategy = get_strategy();
        set_strategy(0x40)

        /* alocira(ne also Inadno(i se ona TSR                             */
        if (allocate(parse |unsigned *) 0x00C_psp) != -1)
                abort();

        /* postavlja slusnu u videoInt  da bi sprecili brisanje bloka      */
        * (short far *)(MK_FP(||unsigned) new_psp - 1  0x01)) = (short) new_psp;

        /* spisime ime TSR u njegov DOS  CODE bl programi tipa  DOS
           Imali vec da replisu s njom                                     */
        fmemcpy(MK_FP(||unsigned) new_psp - 1  0x08)  "TSRtest"  8);

        /* vratimo originalnu strategiju rada da Inoves bloknva            */
        set_strategy(old_strategy)

        /* kopiramo sve ova DOS ||a potrebno za rad novog TSR              */
        _fmemcpy(MK_FP(dos_psp FP_OFF(dos_Timer)  new_timer, Tsr_Size)

        /* premestamo tajmer prekid                                        */
        Setvect(0x1c  |void interrupt (*||Inar)| |new_psp + FP_OFF(new_Timer)||

        /* terminiramo legalno                                             */
        return 0
}
```

Zato moramo u svaku referencu bilo kom entitetu dodati podatak o segmentu u kom radimo što nam omogućava _CS modifikator već objašnjen u prošlom broju i računati položaj entiteta kao ofset od neka granice (u našem slučaju, naravno, od početka CODE segmenta)

Evidentno je da smo ovde ispustili dobar komad objašnjenje i to iz dva razloga prvo da ostavimo čitaocu da ponešto i sam otkrije jer je to najbolji načine da se pronikne u tajne nepoznatog, i drugo, da ostavimo mesta za još neke interesantne stvari koje slede

I tako smo stigli do kraja sa problemom prvim Ako program dat na listingu prevedete (u obliku projekta, recimo) Borlandovim C++ prevodiocu, dobićete jedan 64 bajta dugačak rezidentni program, koji će pri svakom pritisku na desni CTRL taster u desnom gornjam ćošku ekrane ispisati jedan ASCII znak Dakle baš kao što smo rekli glup, ali malecki

PROBLEM DRUGI: KAKO ODREDITI VELIČINU

Problem drugi je jednostavno pitanje „Hteo bih da koristim legalnu funkciju KEEP kako bih neterao moj program da postane rezidentan ali ne znam kako de odredim veličinu programa u paragrafima što funkcije traži kao parameter"

Odgovor se svodi na pažljivo čitanje teksta o memorijskim modalima iz prošlog broja u kombinaciji se ranije stečenim znanjime o strukturi programa u lancu blokove memorije Naime za program znamo da odmah posle njegovog MCB dolazi struktura poznata kao PSP („Program Segment Prefix") u kojoj se čuvaju raznorazni podeci potrebni za funkcionisanje istog što ne služi kao pokazatelj segmentne adrese početka programe Sa druge strane znamo da registar SS čuva početak segmenta gde se krije stak programa (segmentna adresa) a SP trenutni položaj na njemu (u bajtovima) Odatle sledi da će veličina programa biti

(_SS + (_SP/16) – _psp)

gde nam _psp daje segmentu adresu početka PSP-a

PROBLEM TREĆI: NEVOLJE SA BIOSOM

Problem treći se svodi na sledeće pitanje „Previo sam neki svoj rezidentni program koji čak uopšte ne pretuzima DOS prekide nego neki tamo BIOS prekid i on mi zaglavi mašinu Zašto kade je sasvim bezopasan?" Jedan program ovog tipa je dat na listingu jedan, gde preuzimamo prekid 15h (CTRL Break) i pokušavamo da napravimo svoju rutinu koja će u trenutku pritiska odogvarajućih tastera ispisati nešto na ekran, sa čekati pritisak na neki drugi taster, i zatim nastaviti dalje kako i inače treba.

Prvo što čoveku obično padne na pamet, a posle pažljivog čitanja takstа iz prošlog broja, jeste de neka od funkcija koje se koriste u realizaciji nešto brlja po vrednostima registara, i time

zbunjuje mašinu. Naime, rekli smo da DOS prekid ne mogu da se zovu ako se neka od funkcija iz tog skupa već koristi, pa čak i smo u čuvanju stekova ze potrebe DOS-a spasavali podatke o otvorenim datotekama i nigde nismo pomanuli BIOS.

Odgovor je, međutim, jednostavan. BIOS takođe nije predviđan za višeprogramski rad! Naime, u pomenutom listingu preuzimemo jedan BIOS prekid, a u okviru njega poše ispisa poruke korišćenjem drugog BIOS prekida očekujamo pritisak na taster (funkcije GETCH poziva prekid 09h). Efekat je isti kao kod poziva jedne DOS funkcije iz druge, a svodi se na objašnjenje koje smo povodom toga dali u prošlom broju. Dakle, rešenje je da iz naše rutine izbacimo poziv GETCH funkcij

PROBLEM ČETVRTI: NEVOLJE SA VIDEO PREKIDOM

Evo još jednog sličnog pitanja, koje, pak, zahteva mnogo više rasprave. „Pokušao sam da napravim jedan trivijalan rezidentni program koji preuzima *Video* prekid, i iz njega jednostavno poziva originalni prekid. Iako moj TSR zapravo ne radi ništa, po startovanju razidentnog programa ispis na ekran počinja da se čudno ponaša, a ponekad se i potpuno gubi. Gde grešim?" Opet odgovarajući primer je dat na listingu dva.

Da bismo objasnili problem, pogledaćemo kako tačno izgleda preuzimanja prekida u Boland C++ prevodiocu. Na slici dva vidimo kako izgleda jedna prazna funkcija kojom ćemo preuzeti naki prekid u esemblerskoj verziji. Očilo da ovako prazna funkcija redi redundantni posao, jer prvo na stek stavlja sva registra a zatim ih jednostavno vraća. Ako sada u ovu funkciju dodamo poziv originalnog *Video* prekida, situacija postaje kao na slici tri. Pitate se šta bi sad moglo tu da bude loše? Pre pozve funkciji čuvamo sve registre, a po povratku iz nje ih vraćamo na staro. Problem se zapravo, razbija na njih dva parametri koje je neko (avantulano) u registrima prosledio našoj prekidnoj rutini bivaju zbrljani pre nego što stignu do poziva originalnog *Video* prekida (u našem primer. ne osnovu slike tri, vidimo de se gube baš sadržaji OS i BP registare) i parametri koja originalni *Video* prekid vraća po završenom radu se gube skidanjem početnih vrednosti se steka (opet slika tri) ne kraju naša prekidne rutine.

Prenošenje parametara rešavemo na jednostavan način. Borland nudi nešto promenjenu sintaksu prekidna rutina (slika četiri) gde dobijamo mogućnost da početne vrednosti registara tretiremo kao odgovarajuće ulazne parametre. Sa druge strana, ovaj prevodioc nam omogućava i da pristupimo trenutnoj vrednosti registara, konšćenjem predefinisanih „promenljivih" tipa _AX. Slika četiri nudi uprevo ovakvo rešenja ze sličan problem zadatom samo što je u pitanju prekid 13h.

U našem slučaju su stveri za nijansu komplikovanije. *Video* prekid prima parametre i u BP registru, a kod nas, kao što se sa slike tri vidi, ovaj registar dobije novu vrednost ne ulazu u prekidnu rutinu. Zaključak je, dakle, de je potebno urediti dva stvari pre poziva originalnom *Video* prekidu naći parametre koji su mu bili namenjeni, e po povratku iz njega u našu rutinu osigurati se da povratne peremetra ne izgubimo. Konačno rešenja je dato na listingu tri.

PROBLEM PETI: VIDEO NA LAKŠI NAČIN

-je proizašao iz prethodnog. „Postoji li neki lakši način da sa pozova originalni *Video* prekid bez svih onih žonglirarije sa registrima?" Istovre meno, ovo je i odgovor ne pitanje u vezi tekstra iz

Listing 1

```
#include <bios.h>
#include <dos.h>
#include <conio.h>
#include <alloc.h>

void interrupt newint08(void)
{
    inc row, col,

    disable();
    row = wherey();
    col = wherex();
    gotoxy(1,1);
    cputs("Ovo je odstampano sa TSR!!! ");
    gotoxy(row, col);
    enable();
}

void main()
{
    setvect(0x1c, oldint1c);
    keep(0 ,(_SS+(_SP/16) -_psp));
}
```

Problem sa BIOS prekidom

Listing 2

```
#include <dos.h>
#include<alloc.h>

void interrupt far *oldvideo(void);

void interrupt videoint(void)
{
    oldvideo();
}

void main()
{
    oldvideo=getvect(0x10)
    setvect(0x10 ,videoint);
    keep(0 ,(_SS+(_SP/16) _psp));
}
```

Problem sa VIDEO prekidom

prošlog broja „Čemu služi funkcije _CHAIN_INTR?"

Odgovor na prethodni problem ja proizašao iz pretpostavka da se posle poziva originalnog *Video* prekida treba vratiti u našu prekidnu rutinu da bi se tamo uradio još neki posao. Zbog toge smo prvo morali pronaći sve perametre koji su se slali originalnom prekidu a zatim paziti da na izgubimo parametre koja on vreća. Međutim, ako malo pažljivije pogledamo sliku tri, uočićemo da su svi pozivni parametri već sačuvani na steku. Bilo bi zgodno kada bismo mogli da, pra poziva, skinemo sa steka sve te vrednosti i prosledimo ih originalnom prekidu. Tada bi se, međutim, po povratku iz njega na kraju naše rutine pokušalo sa skidanjam tih istih parametara sa steka a tamo ih više nemal. Konačno, to je i ideja. možde bismo mogli da akinemo početna vrednosti registara sa staka, e onda bespovratno skočimo na originalni *Video* prekid, tako da se ovaj direktno vreti tamo odakle je dosao ukupni poziv našoj rutini, i direktno preda povratne parametre.

Jedno rešenje je dato slikom pet, gde činimo upravo ono što smo i zamislili. Imutak pre nego što treba da skočimo na originalni prekid pravi mamo sve originalna vrednosti sa steka, a ne njega stavljamo adresu skoka. Jedno RETF na kraju će sa steka skinuti tu adresu, što će rezultoveti bezuslovnim skokom. Lakši način bi bio korišćenje legelna funkcija _CHAIN_INTR, koje čini upravo ovo, e kao parametar uzima uprevo adresu prekidne rutina na koju se skače.

Ne ovaj način se eliminišu oba podproblema iz prethodnog problema, ali uz ograničenje da posle ovog postupke više neme povratka u našu prekidnu rutinu (što poziv originelnom prekidu efaktivno postavlje na kraj cele procedure).

Napomena: Za sve primere iz ovog teksta korišćen je Borland C++ prevodilac. Autor ne garantuje da će neki drugi prevodilec moći previlno de preveda priložene primere, niti da će sva razmetranja iz teksta važiti i ze njega.

STA IMA NOVO

HARDVER

Računari

Počinje nova moda: Ekološki računari

„Dell Computer" se pridružuje sve većoj listi ekološki svesnih proizvođača koji poštuju preporuke Američkog zavoda za zaštitu životne sredine. Za računare, to znači da troše manje od 30 vati u neaktivnom stanju i tada mogu da se zaključe ituiom „Energy Star" koju ovo društvo dodeljuje. Američka vlada procenjuje da se na taj način može uštedeti 40 miliona dolara godišnje, što doprinosi i smanjenju zagađenja koje nastaje pri proizvodnji električne struje.

Sve je to lepo, ali kupci su spremni da vode računa o ekologiji samo ako to ne izikuje dodatene izdat ke u njihovog džepa. Štedljivi računar 425e/L sa fiopidrajvom, tvrdim diskom od 120 MB, 4 MB sistemske memorije mišem instaliranim DOS-om i Windows-om ima cenu od 1 270 dok će računar slične konfiguracije 466/L, sa tvrdim diskom od 170 MB koštati 2.150 dolara. Rekle bi se da dodaci koji obezbeđuju štednju struje nisu uticali na povećanje cene.

„Dell" razmišlja o uvođenju i drugih tehnologija za uštedu energije kao što su ravni ekran, PCMCIA uređaji i još neka ergonomska rešenja.

Kontakt adresa: Dell Computer Corporation, 800-289-3355 ili 512 728-4400)

Tvrdi diskovi

NEC odustaje od 2,5 inča

NEC je nameravao da u martu 1994. ponudi na tržištu 2.5-inčne tvrde diskove kapaciteta 265 MB ali je od toga odustao zbog male potražnje. Kažu da nisu uspeli da nađu ni jedtine delove potrebne za proizvodnju. Konkurencija na tržištu 2.5-inčnih diskova je žestoka, jer mnoge firme nude veoma slične proizvode, tako de se neki povlače (nedavno je to učinila kompanija Japan Victor).

Umesto 2.5-inčnih NEC planira da krajem godine ponudi 1.8-mčne tvrde diskove sa kapacitetom od 125,8 MB, koji će biti izuzetno malih dimenzija i težiti samo 75 grama. Cena će iznositi samo $450, zahvaljujući smanjenim troškovima proizvodnje.

Adresa: NEC, +81-3-3451-2974 Fax, +81-3-3457-7245

Monitori

Usparni monitor: Štedite struju, ona je skupa

Na izložbi PC Expo koja je početkom jula održana u Nijujorku, „Zenith Data Systems" je predstavio novi monitor koji se sam uspava tokom perioda neaktivnosti naravno radi štednje struje.

15-inčni Super VGA *multisync* displej koji na treperi daje rezoluciju do 1024x768 piksela i namenjen je grafičkim aplikacijama, stonom izdavaštvu, projektovanju III drugim okruženjima gde korisnik dugo gleda u ekran. Ovaj monitor, nazvan ZCM 1540, automatski prelazi u režim smanjene potrošnje struje kada mije u upotrebi. Svaka aplikacija koja ima sposobnost zamračenja ekrana, kao *Microsoft Windows*, i svi ene mnogobrojni „screen saver" programi automatski prebacuju monitor u *stand-by* režim.

Potrošnja se smanjuje za 30 posto, što znači de troši manje od 30 vati u *stand-by* režimu, čime konkuriše za titulu „Energy Star" standard za proizvođača kompjuterske opreme koji je ustanovilo Američko društvo za zaštitu životne sredine.

Nakon izvesnog vremena provedenog u *stand-by* režimu, ZMC 1540 odlazi na odmeranje (*rest state*) kada troši samo 5 vati. Pritisak na bilo koji taster vraća monitor u budno stanje. Monitor ima raster 0.28, automatsku sinhronizaciju, sam otkriva napon napajanja, pruža podršku za pet različitih rezolucija i odgovara i američkim i evropskom (posebno strogim) standardima o zaštiti od zračenja.

Cena će biti u rangu ostalih 15-inčnih monitora.

Kontakt adresa: Zenith Data Systems, 800-553-0331)

ŠAHOVSKA PITALICA

Posle dužeg vremena imali smo jednu pitalicu kod koje zaista nije bilo teško izabrati pobednike – iako je u predviđenom roku pristigio 64 odgovora, za nagrade su konkurisala tačno tri kupona!

Možda je razlog za takvu selektivnost ovoga problema bila njegova priroda – pitalica je bila više šahovska nego računarska: da je objevljena u nekom šahovskom glasniku svakako bi bilo više optimalnih rašenja. Bilo kako bilo, za početak ćemo se podsetiti problema. Osim „klasičnih" poteza pravila šahovske igre obuhvataju i dve neobične stvari – rokadu i uzimanje „u prolazu" *en passant*. Rokade je jedini potez u kome se pomeraju **dve** figure – kralj dva mesta prema topu, a top do kralja pri čemu postoji male (preme bližem topu) i velika rokada. U nekim izuzetnim situacijama (do kojih se, varovatno nikada nije došlo u nekoj pravoj šahovskoj partiji) rokadom se protivnik može i matirati.

Uzimanje „u prolazu" obavlja se pionom na četvrtom redu. Recimo da je crni pion na g4 a beli odigra h2-h4 „preskačući" pri tome polje h3. Crni tada ima pravo da tog pešaka pojede pri čemu njegov pion dolazi na h3 baš kao da je beli igrao h2-h3 a ne h2-h4. Ukoliko crni od**mah** ne uzme pione sa h4 gubi pravo na *en passant* uzimanje u sledećim potezima. Teorijski gledano, *en passant* uzimanjem se može dati i mat, s tim što je ta teorija poneki put prerasla i u prekaz – par čitalaca nam je poslalo reference na šahovske partije koje su zaista završene *en passant* matom, obe reference se odnose na prilično davne turnire, a za aktere tih partija verovatno su čuli samo najpredanliji istončari šaha.

Trebalo je iskomponovati poziciju koja bi sjedinjavala mat rokadom i *en passant*-om – trebalo je da beli na potezu deje mat vršeći rokadu a crni na potezu treba nekim *en passant* uzimanjem. Da bi sve bilo još malo teže zahtevali smo da ne postoje alternativni matovi u jednom potezu ni za belog ni za crnog – nedopustivo je recimo da beli umesto male rokade odigra Ke1-e2 i tako da mati. Rešenja su oenjena po broju figura (računajući tu i pione) na tablu – što je figura manje to je rešenje bolje. Trebalo je poštovati i previlo šahovske igre (koje nismo epomenuli u postavci) da je rokada dozvoljena samo eko ni jedno polje preko koga pralazi kralj nije napadnuto protivničkom figurom (znači ni polje na kome se kralj nalazi pre rokade ni dva polja desno odnosno levo od njega) za polje na kome se nalazi top nema tog ograničenja.

Slika 1: Pozicije figura koje su fikalrane problemom

Slika 2: Skiciran mat belog

Slika 3: Konecno rešenje problema

Pošto beli treba da matira crnog rokadom, beli kralj se mora nalazrti ne E1 (početna pozicije). Da bi crni mogao da matira belog *en passant* potezom, taj potaz mora otvoriti bilo dve dijagonele, bilo jednu liniju i jednu dijagonalu. Pretpostavimo de je beli top na h1 i da beli može da izvrši malu rokadu. Pošto tada polja e1 f1 i g1 ne smeju biti napadane *en passant* potez crnog mora biti takav da otvara napad na polja e1 i f1 (spostavlja se da su ovi uslovi ispunjeni samo eko crni otvara dve dijagonale koje mogu biti napadane bilo sa dva crna lovca bilo lovcem i damom. na primer kao na slici 1.

Potezom crnog c4xb3 oslobađaju se dijagonale oba crna lovca, i da bi beli bio matiran treba pokriti i polja F2 i D1. Da bi mat rokadom bio jedinstven crni kralj se mora nalaziti u prvom redu – ako bi se crni kralj nalazio negde na F liniji bio bi moguć mat rokadom, ali bi bio moguć i mat potezom Tf1. Najzgodnije je postaviti crnog kraljа na c1 i time napasti i polje d1 koje nam treba za mat crnog. Da bi se kompletirao mat belog treba negde na drugom redu postaviti bilo belog topa ili belu damu. Polje f2 zbog mata crnog treba da bude zauzeto pa je najzgodnije baš tu postaviti belog topa. Teko dolazimo do slike 2.

Ostaje još samo da se dovrši mat crnog treba se pobrinuti za „opasnost" od iskoraka belog kraljа na f2 posle *en passant* uzimanja što se svodi na postavljanj belog topa u drugi red. No taj top ne može biti bilo gde da se ne bi dogodilo da beli zakloni kralje od napada lovce. Zato ćemo topa

KOLIKO BITA PO POZICIJI?

85. Pitalica je zapravo bila uvod u 87. koja će se takođe baviti šahom, ali na način koji će možda više odgovarati programerima. Videli smo da pozicija na šahovskoj tabli nije strogo statična stvar – u datom primeru, recimo, podatak da li crni ima ili nema pravo na *en passant* uzimanje o beli na rokadu predstavlja razliku između pobede i poraza. Vaš zadatak je da odredite koliko je **najmanje** bita potrebno da bi se opisala proizvoljna šahovska pozicija.

Podrazumeva se da je pozicija regulerna tj. da se do nje moglo doći normalnim tokom šahovske partije – ne dolaze dakle u obzir pozicije bez kralja, sa nekoliko belih kraljeva, sa više od 8 crnih pione ili sa 10 crnih dama i tome slično.

Potrebno je da uz poziciju budu upisane i sve relevantne informacije ko je na potezu, da li svaki od igrača ima pravo rokade na veliku ili na malu stranu (ako kralj nije na početnoj poziciji, ovaj podatak se ne mora čuvati), da li svaki od piona na petom odnosno četvrtom redu ima pravo da *en passant* uzme susednog piona protivničke boje i tome slično. Kratko rečeno, svi podaci neophodni da se partija nastavi od „zaleđenog" mesta bez potrebe da se čuvaju podaci o „istorijatu" zatečene pozicije. Potrebno je da metod radi za **sveku** poziciju a broji se koliko bita je **najviše** potrebno za poziciju koja je za dati metod, „najkomplikovanija".

KAKO DOSTAVITI REŠENJE

Rešenja pitalice zajedno sa **obrazlozenim** programima koje ste napisali pošaljite na uobičajenu adresu „Računari" (za Dejanove pitalice). Bulevar vojvode Mišića 17, Beograd tako da pristignu **pre** 20. septembra 1993. Sva pisma sa korektnim rešenjima konkurišu za novčane nagrade u iznosu cene 10 7 odnosno 5 primeraka „Računara" dok će kuponi (ili njihove fotokopije) na koje je upisan identifikacioni broj učestvovati u goriš'njem takmičenju rešavača pitalica. Identifikacioni broj dobijate tako što u prve tri kućice upišete **poslednje** tri cifre nečijeg broja telefona u sledeće dve godinu vašeg rođenja i na kraj dodate dve cifre po izboru. Obratite pažnju da sva rešenja koja šaljete u toku godine nose isti identifikacioni broj. Osim običnom rešenje ove pitalice možete da pošaljete i elektronskom poštom posredstvom Sezama – koristite menije REDAKCIJA, PITALICE, REŠENJE ili iz komandnog moda pošaljite ličnu poruku korisniku pitalice (*mail wnts pitalice*).

87: Za "pamćenje" proizvolje pozicija potrebna su _____ bita

Ime i prezime _____

Adresa _____

Mesto _____

Kompjuter _____ Vreme _____

Identifikacioni broj |_|_|_| |_|_|_| |_|_|_|
iz broja tel. | god. rod. | po izboru

„maskirati" crnom damom na h4 čime dolazimo do konačne pozicije sa slike 3

Pored ova pozicija moguće su još dve varijante crna dama se postavlja umesto crnog lovca na a5, a crni lovac na g3 ili h4 (pozicija sa crnom damom na g3 ne ispunjava uslove jer ja u tom slučaju napadnuto polje g1, pa nama ništa od rokada belog) Simetrična pozicija sa crnim lovcima na desnoj *atrani* nija moguće zato što se na može otvoriti napad na belog kralja Iz tih razloga nija moguća (pogotovu sa devet figura) sastaviti poziciju u kojoj beli matira velikom rokadom

Vredi reći da rešenja sa slika 3 lako potpuno ispunjava uslova zadatka na bi bilo priznato kao šahovski problem jer je po kodeksu Staine komisije FIDE za šahovsku kompoziciju (na koji nam je skrenuo paznju **Milan Vallmirović** (*mivel* na Sezamu)) *en passant* moguć samo ako postoje egzaktan dokaz da je protivnik u prathodnom potezu napravio iskorak pionom Što sa tiče rokada.. ona je dozvoljena ako se do pozicija na dijagramu moglo doći bez pomeranja kralje I topa u toku dokazna partija Ova dva pravila su po našem skromnom mišljenju, u neskladu jedno sa drugim tj nija jasno zašto se na bi *en passant* tra tirao kao i rokada ili obrnuto no tako je kako je Teško je zamislivo da bi se problem mogao rašrti ćek i sa više figura, uz poštovanje ovog ograničenja (mada ko zna?) tako da sa bojimo da ća ostati u društvu Fišera i Kasparova – nepriznat od FIDE

U predviđenom roku primili smo četiri koraktna (i praktično identična) rašenja sa devet figura na tabli pri čemu su samo tri učestvovala u izvlačenju – **Milan Vallmirović** je izričito tražio da njegov odgovor ostana „van konkurencije" Najviše sreće imao je **Milutin Smiljanić** iz Beograda koma pripada prva nagrada u vrednosti 10 pnmeraka „Računara" Drugu nagradu (vrednost 7 primeraka „Računara") dobio je **Stojan Miloradović** iz Beograda (*smiloradovic* na Sazamu) a treću (vrednost 5 pnmeraka „Računara") **Dragan Petrović** iz Novog Sada Za godišnja takmičanja bodovaće se kao tačna i rašenja sa 10 i 11 figura na tabli velika većina pnmljenih pisama spada u ovu katagonju mada je bilo i rešanja sa 14 i više figura

NOVOSADSKI SAJAM
Specijalizovane Septembarske Manifestacije

27.09. - 02.10.93.

Poslovni USPEH stanuje na ovoj adresi

NOVOSADSKI SAJAM DD
Hajduk Veljkova 11
21000 Novi Sad

Tel: (021) 51-570, 51-977
Fax: (021) 616-121

Međunarodni Sajam Elektronike, Telekomunikacija i Procesne Opreme

Međunarodni Sajam Informatike

INFORMATIKA

ELEKTRONIKA

COMPUTRONIC
Preduzeće „MZ" 11000 Beograd
Dubljanska br 70 Tel 011/434-812
Fax 011 450 471

ŠTAMPAČI
EPSON

LX-400, 9-pin A4 format, 180 z/s, ćirilat
LX-100, 9-pin A4 format, 240 z/s, CSF, YU-set
LX-850, 9-pin A4 format 200 z/s YU-set
FX-870, 9-pin A4 format 380 z/s, YU-set
LX 1050, 9-pin A3 format, 200 z/s, YU-set
FX-1050, 9-pin A3 format, 300 z/s, ćirilat
LQ-100 24-pin A4 format, 200 z/s, CSF ćirilat, ESC P-2
LQ-570, 24-pin A4 format 240 z/s, YU-set ESC P 2
LQ-870, 24-pin A4 format, 330 z/s, YU-set ESC P 2
LQ-870 24-pin A4 format, 300 z/s YU-set kolor
LQ-1070, 24-pin A3 format, 240 z/s YU-set ESC P 2
LQ-1170, 24-pin A3 format, 330 z/s, YU-set, ESC P-2
LQ-1060, 24-pin A3 format, 300 z/s YU-set, kolor
DLQ-2000, 24-pin A3 format, 270 z/s, YU-set, kolor
DFX-5000, 9-pin A3 format, 533 z/s, YU-set
DFX-8000, 18-pin A3 format, 1066 z/s, YU-set
SQ-870, 48-mlaz, A4 format, 960 z/s, YU-set, INK JET
SQ-1170, 48-mlaz, A3 format, 960 z/s, YU-set, INK JET
EPL-4000, LASER, 300 DPI, 6 s/m 1 MB, H-YU-set, (opcija)
EPL 4300 LASER 300 DPI 6 s/m 1 MB, RITech, H-YU-set (opcija)
EPL-8100, LASER 300 DPI, 10 s/m 2 MB RIT PCL 5 H-YU-set, (opcija)
EPL-7500 POSTSCRIPT LASER MEGA2 300 dpi 6 s/m 2 MB
GT-8000 COLOR IMAGE SCANER A4 format, 800 DPI

Roland DIGITAL GROUP

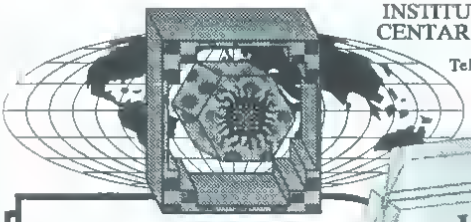
DXY 1100 A3 format, 8 pera, 42 cm/s, mag držanje papira
DXY 1200 A3 format, 8 pera 42 cm/s ele statičko papira
DXY-1300 1MB A3 format 8 pera 42 cm/s ele statičko papira
SKETCH MATE A3 format 8 pera
SKETCH MATE A4 format 8 pera
DPX 2500 A2 ploter tabla 8 pera 62 cm/s 1 MB
DPX-3500 A1 ploter tabla 8 pera 62 cm/s 1 MB
GSX-9000 A1 „ROLL"-ploter 8 pera 113 cm/s 1 MB aut. p v p
GRX-300 AG A1 „ROLL" ploter 8 pera 66 cm/s
DPX-4600 A0 ploter tabla 8 pera, 62 cm/s 1 MB
GSX-4000 A0 „ROLL" plotar 8 pera, 113 cm/s 1 MB aut. p v p
GRX-400 AG A0 „ROLL"-plotar 8 pera, 66 cm/s
LTX 100 A3 format termički ploter
LTX-120 A3 format termički ploter
LTX 321 A1 format termički ploter
LTX-420 A0 format termički plotar
Data buffer SYA 550
CAMM-1 PNC-1100 PLOTER-REZAČ format od 50 do 640 mm (do 1600 mm)
CAMM-1 PNC 1600 PLOTER REZAČ A0 format (do 3600 mm)
STIKA Rezač sa ugrađenim skenerom sa RS232C i programom

EIZO

EIZO 9080i 16" kolor monitor 1024x768
EIZO 6500 21" monohromatski monitor 1684x1200 (60Hz) 1280x1024
EIZO F550i 17" kolor mon ravan ekran max 1280x1024
EIZO F750i 21" kolor mon ravan ekran max 1280x1024
EIZO T560i 17" kolor (TRINITRON CRT) 1280x1024 CRT Trio p 0:31
EIZO T660i 20" kolor (TRINITRON CRT) 1280x1024
EIZO GRAFIČKE KARTICE – VA30 VA41 AA51 MD B09 MD B10 MD B12

GARANCIJA GODINU DANA
POSEDUJEMO DODATNI I POTROŠNI MATERIJAL
ĆIRILIČNO-LATINIČNI EPROMI ZA LX,FX,LQ
MODELE OVLAŠĆEN SERVIS
INFORMACIJE NA TEL 011 434 812 FAX. 011 450 471

INSTITUT ZA NUKLEARNE NAUKE "VINČA"
CENTAR ZA PERMANENTNO OBRAZOVANJE
11000 Beograd, Nemanjina 4/IX
Telefon: 011 / 683-390, 682-486, 641-155 / 107, 181
Telefax: 011 / 682-486



BIBLIOTEKA INFORMATIKA

1. **AutoCAD** (verzija 11)
konstruisanje i projektovanje pomoću personalnih računara
Autori: Bora Damjanović i Petar Damjanović
Sedmo izdanje, izlazi iz štampe u septembru '93.

2. Uvod u **C jezik**
Autor: Vladan Vučić
Četvrto izdanje, 1991. - latinica, 330 strana, format B-5

3. Prenos programa **SYMPHONY** na personalnim računarima
Autor: Dragan Pantić
Treće izdanje, 1990. - latinica, 232 strana, format B-5

4. **OS / 2** - vodič za korisnike
Autor: Zorica Jelić
Drugo izdanje izlazi iz štampe u decembru '93.

5. **VENTURA** - računarsko izdavaštvo
Autor: Predrag Davidović
Treće izdanje izlazi iz štampe u decembru '93.

6. **FORTRAN 77**
standard sa dopunama za personalne računare
Autori: Vlajko Kecić i Zoran Konstantinović
Treće izdanje, 1993. - latinica, 436 strana, format B-5

7. **UNIX** - vodič za korisnike
Autor: Zorica Jelić
Treće izdanje, 1991. - latinica, 436 strana, format B-5

8. Prenos programa **FRAMEWORK III** na personalnim računarima
Autor: Dragan Pantić
Drugo izdanje, 1991. - latinica, 330 strana, format B-5

9. **Programski alati u matematici**
MathCAD, GRAPHER, EUREKA
Autor: Ante Ćurlin
Drugo izdanje izlazi iz štampe u decembru '93

10. **QUATTRO PRO 4**
Autor: Dragan Pantić
Drugo izdanje, 1993. - latinica, 361 strana, format B-5

11. **DOS UKRATKO** (verzije 3.3, 5.0 i 6.0)
Autor: Dragan Pantić
Treće izdanje, izlazi iz štampe u junu '93

12. Vodič za **VAX / VMS**
Autori: Tamaš Kerepeši, Zvonko Oršolić i Saša Malbaški
Prvo izdanje, 1990. - latinica, 532 strana, format B-5

13. Prenos programa **EXCEL** na personalnim računarima
Autor: Dragan Pantić
Prvo izdanje, 1991. - latinica, 278 strana, format B-5

14. **UNIX** - vodič za programere
Autor: Zorica Jelić
Prvo izdanje, 1991. - latinica, 345 strana, format B-5

15. **WINDOWS 3.1**
Autor: Dragan Pantić
Drugo izdanje, 1993. - latinica, 349 strana, format B-5

16. **PRIMAVERA**
upravljanje projektima uz pomoć računara
Autori: Jaroslav Urošević i Jelica Dražić - Ostojić
Prvo izdanje, 1991. - Latinica, 368 strana, format B-5

17. **dBASE III+** priručnik
Autor: Milorad Filipović
Prvo izdanje, 1991. - latinica, 254 strana, format B-5

18. **Osnovi informatologije i informacione tehnologije**
Autor: Ljubomir Dukić
Prvo izdanje, 1991. - latinica, 542 strane, format B-5

19. **LOTUS 1-2-3** (verzije 3.0 i 3.1)
Autor: Dragan Pantić
Prvo izdanje, 1991. - latinica, 418 strana, format B-5

20. **dBASE IV** priručnik
Autor: Ljubomir Lazić
Prvo izdanje, 1991. - latinica, 318 strana, format B-5

21. **WORDPERFECT** (verzija 5.1)
Autori: Dragan Pantić i Nada Pantić
Prvo izdanje, 1991. - latinica, 304 strane, format B-5

22. Programiranje u **CLIPPER-u 5.01**
Autor: Alempije Veljović
Drugo izdanje izlazi iz štampe u junu '93.

23. **FoxPro**
Autor: Dušan Čalić
Prvo izdanje 1992. - latinica, 403 strane, format B-5

24. **Uvod u strukture podataka**
Autor: Miroslav Jeckovć
Prvo izdanje, 1992. - latinica, 358 strana, format B-5

25. **ORACLE** (verzija 5) - arhitektura i administracija
Autor: Vladimir Mihajković
Prvo izdanje, 1992. - latinica, 180 strana, format B-5

26. **WORD za WINDOWS**
Autori: Dragan Pantić i Nada Pantić
Prvo izdanje, 1992. - latinica, 320 strana, format B-5

27. **MS WORKS** - upravljanje poslovnim procesima
Autori: Mirjana Nikolić i Miroslav Nikolić
Prvo izdanje, 1992. - 290 strana, format B-5

28. **PASCAL**
standard sa dopunama za personalne računare
Autori: Zoran Konstantinović i Slobodan Simić
Prvo izdanje, 1992. - latinica, 401 strana, format B-5

29. **OPERATIVNI SISTEMI**
MS DOS 3.30 do 6.0, NDOS 6.0 i 6.01, DR DOS 6.0
Autori: Dragan Cvetković i Redica Cvetković
Prvo izdanje, 1993. - latinica, 266 strana, format B-5

30. **LOTUS 1-2-3** (verzije 2.3) i modeli za poslovno odlučivanje
Autori: Mirjana Nikolić i Miroslav Nikolić
Prvo izdanje, 1993. - latinica, 227 strana, format B-5

31. **PC PAINTBRUSH IV PLUS**
(WINDOWS 3.1 APPLICATION)
Autor: Miodrag Delaković
Prvo izdanje, 1993. - latinica, 148 strana, format B-5

32. **UVOD U SISTEMSKO PROGRAMIRANJE III**
Autor: Dušan Velašević
Prvo izdanje, 1993. - latinica, 618 strana, format B-5

PORUDŽBENICA — na stranji uz redni broj knjige upisati broj poručenih primeraka

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

IME I PREZIME
(Naziv preduzeća) _____
ULICA I BROJ _____
BROJ POŠTE I MESTO _____
TELEFON _____

20 21 22 23 24 25 26 27 28 29 30 31 32

"PERSONALNI računari", juli 1993.

Poručbenicu podeli sa petim primerkom upletnice (informacije o cenama svakog radnog dana od 8 do 20 sati)

Uplata se vrši na žiro račun
INSTITUT "VINČA", LABORATORIJA 060, broj 60803-609-698.
Troškove isporuke knjiga snosi poručilac i namiruje ih unapred zajedno sa uplatom za knjige

694 [MATS] Verovatno ste u dnevnoj štampi pročitali vest da je posle više vekova dokazana poslednja Farmaova teorema. Dnevna štampa kao što to obično biva, retko navodi ikakve tehničke detalje pa verujemo da će mnoge naše čitaoce koji se interesuju za matematiku zanimati kratak pregled zasnovan na brojnim porukama sa stranih mreža koje su ovim povodom prenesena na Sezam.

Poslednja Farmaova teorema ili u engleskoj literaturi FLT kaže da se ne mogu naći prirodni brojevi *a*, *b* i *c* takvi da je $a^n+b^n=c^n$ za n>2 (za n=1 problem je trivijalan dok za n=2 postoji beskonačno mnogo rešenja koja često nazivamo Pitagorinim brojevima, recimo a=3 b=4 i c=5). Sve je počelo kada je Farma na marginama knjiga napisao (u slobodnom prevodu – originalni latinski tekst možete naći na Sezamu): *za druge strane nemoguće je predstaviti kub kao zbir dva kuba ili četvrti stepen kao zbir dva četvrta stepena ili uopšte bilo koji stepen osim drugog kao zbir dva stepena sa istim eksponentom. Pronašao sam zaista zadivljujući dokaz ove tvrdnje koji na žalost, ne može da stane u ovu marginu.*

Nikada nečemu znati da li je Farma zaista imao pa umu ispravan dokaz, pošto takav nije nađen među njegovom zaostavštinom. Problem je međutim zahtevao matematičare čitavog sveta koji se već vekovima trude da dokazu ovu naoko sasvim jednostavnu tvrdnju ili da nađu kontraprimer koji bi pokazao da tvrdnja nije tačna. Pokazalo se da je kao dokaz zadnje duboko u tajne dolasti numeričke teorije tako da je tek u poslednjih dvadeset godina učinjen ozbiljniji pomak na planu dokazivanja. Glavnim prodorom se smatra Faltingsov dokaz Mordelove hipoteze koja slobodno formulisano, kaže da algebarska kriva roda 2 ima najviše konačno mnogo racionalnih nula. Kako je Faltings dokazao da je to tačno za krivu $u^n+v^n=1$ za n>3 sledi da za neko konkretno *n* Fermaova jednačina ima najviše konačno mnogo rešenja. Trebalo je dakle još dokazati da ih rešenja uopšte nema.

Pa konačnog dokaza *Wagstaff i Tanner Granville i Monagan i Coppersmith* pokazali su da jednačina nema rešenja za *n* manje od $7.5*10^{17}$ (ovo je inače nezamislivo veliki broj). Takoda je pokazano da za beskonačno velika klase brojeva *n* jednačina nema rešenja. Ali dokaz (doduše indirektni) da jednačina nema rešenja ni za jedno *n* deo je 40 godišnjeg engleski matematičar Endrju Vajls (*Andrew Wiles*). Njegov dokaz se između ostalog zasniva na rezultatima brojnih matematičara posebno na radovima koje su objavili *Flach Hida Kolyvagin Mazur Ribet i Rubin* kao i na ranijim radovima samog dr Vajlsa. Zapravo, Vajls je dokazao Tanijamino (*Taniyama*) hipotezu (da se sve eliptičke krive mogu parametarizovati) za jednu klasu semistabilnih eliptičkih krivih nad Q što kako je ranije dokazao *Ken Ribet*, implicira tačnost poslednje Fermaove teoreme. Za izlaganje čitavog dokaza bilo bi potrebno najmanje 300 strana i kaže se čemu za dodatnu diskusiju i razna pomoćna tvrđenja treba daljih 1000 ili više strana. Dokaz je osim toga tako jak da ga čak i u najgrubljoj meri mogu razumeti samo matematičari i to svakako ne svi matematičari već oni koji se veoma ozbiljno bave teorijom brojeva. Bilo kako bilo teško da je to ono što je Farma imao u vidu napisavši „pronašao sam *zaista zadivljujući* dokaz" – ili postoji neki mnogo jednostavniji dokaz ili Ferma naprosto nije bio u pravu!

Kada se radi o ispravnosti dokaza, većine slušalaca Vajlsovih predavanja na kojima je čitav rad izložen bila je lascinkana uredanim poslom ali to i dalje na znači da se dokaz može bezuslovno prihvatiti. Biće potrebno oko pet nedalja da nekoliko vrhunskih stručnjaka za ovu oblast „pročešljaju" Vajlsov dokaz, a onda možda i pai meseci pre nego što on bude potpuno prihvaćen kao ispravan. Ukoliko sa u njemu nađu neke manjkavosti ili ne preciznosti verovatno je da će sam autor ili neki drugi matematičari uspeti da revidiraju „sumnjiva" mesta dolazeći tako do finalnog dokaza – poznato je na primer da je *Faltingsov* dokaz prošao kroz nekoliko revizija dok nije konačno doveden u red. Kratko rečeno, ne može se još sa sigurnošću tvrditi da je poslednja Farmaova teorema dokazana ali postoji prilična verovatnoća da je tako.

Pomenimo najzad, i komentar Dejvida Karabela (*David Carabello*) sa *Princeton* univerziteta, koji tvrdi da izloženi dokaz na može biti ispravan, jer je prateći „Zvezdana staze – sledeća generacija", čuo „svojim ušima" da poslednja Fermaova teorema nije dokazana ni u njihovo vreme pa samim tim nije mogla biti dokazana ni u dvadesetom veku!

695 [COMM] Iz prima koje je *Th Irmer* direktor telekomunikacionog biroa za standarde u Ženevi uputio direktoru našeg PTT-a proizilazi da je u pripremi promena pozivnih brojeva za državne koje su nastale raspadom bivše Jugoslavije. Očigledno je uložen trud da promene bude što „bezbolnija". Broj 38 koji je do sada karakterisao Jugoslaviju povlači se iz upotrebe. Ubuduće će pozivni broj za SR Jugoslaviju biti 381 za Bosnu i Hercegovinu 387 za Hrvatsku 385 za Sloveniju 386 a za Makedoniju 389.

Telefaks kod 62 koji je nekada pripadao SFRJ ubuduće će koristi SR Jugoslavija dok će telefaks kodovi Bosne i Hercegovine Hrvatska, Slovenije i Makedonije biti 600 599 598 i 597 respektivno. Nije precizno specificiran datum kada će ova izmene stupiti na snagu ali je očigledno da će to biti veoma skoro. Ukratko ako iz sveta ne možete da dobijete broj +38 11 648422 pokušajte sa +381 11 648422 [Preuzelo se Sezama, prilog **Dragan Jovičević** (*djovicevic*)]

696 [DOS,MSDOS 660, PC] Tokom instalacije DOS-a 6.0 kao što smo pisali IO.SYS proverava poslojanje datoteke DBLSPACE.BIN u *root* direktorijumu i aventualno učitava taj program kako bi ubuduće mogao da pristupa segmentima kompresovanog diska. Na taj način je poništeno oko 45 kilobajta osnovne memorije – nema nista od podizanja u *high RAM* jer u trenutku učitavanja samog DBLSPACE.BIN QEMM ili EMM386 nisu učitani pa *high RAM* i ne postoji. Ali stvari se mogu ispraviti editovanjem datoteke CONFIG.SYS i zamenom njene linije u kojoj se pominje DBLSPACE.SYS sa

```
DEVICE=C:\DOS\DBLSPACE.SYS /MOVE
```

I to je sve – DBLSPACE.BIN će biti prekopiran u visoku memoriju oslobađajući tako dragocenih 40-tak kilobajta osnovnog RAM-a.

Pomenuti drajver DBLSPACE.SYS rekli bi sa nije radi ništa – dodeljena mu je samo uloga „lagalizacije" prisustva DBLSPACE.BIN u osnovnoj memoriji. Ukratko, DBLSPACE.SYS će samo formalno zauzeti memoriju koju je DBLSPACE.BIN već konstatovao tako da će DOS prihvatiti memorijsku mapu koja je zapravo „falsifikovana" [Preuzeto iz dokumentacija firma *Quarterdeck*]

697 [OS,MSDOS, PC] Koliko ste puta kucali DIR > A:BAT zatim editovali A:BAT i ubacivali poziv nekog potprograma koji će jednu po jednu obrađivati sve datoteke u tekućem katalogu? Ako to često radite znate koliko je gnjavaža brisati datume i vremena spajati ime i ekstenziju zamenjivati blankove tačkom prosto bi čovek seo da napiše program koji to automatizuje. A nije potrebno – dosta je da umesto DIR koristite DIR /B na primer DIR /B > A:BAT. Ddoicele bas ono što vam je potrebno spisak datoteka bez ikakvih suvišnih podataka i sa tačkom između imena i ekstenzije!

Podrazumeva se DOS 5.0 ili 6.0 [Preuzeto sa Sezama, prilog **Emil Matic** (*ematic*)]

698 [OS,MSDOS, PC] Iako pompezno najavljeni *MemMakar* iz DOS-a 6.0 ne obavlja baš sve ono što bi trebao da obavi. Primećeno je pre svega, da *MemMaker* optimizuje samo jedno konfiguraciju sistema – ako ste koristili novilele DOS-a 6.0 definisali više konfiguracija moraćete više puta da startujete MEMMAKER i prolazite kroz čitavu proceduru višestrukog reselovanja računara. Potencijalno još nepriatnija *MemMakar* na uzima u dozir CALL naredbe iz datoteke AUTOEXEC.BAT – ako se pri podizanju sistema pozivaju potprogrami optimizacija neće uspeti. S obzirom da QEMM-ov OPTIMIZE korektno trebra ova datoteka može se desiti da prelazak sa QEMM-a na EMM386 onemogući podizanje sistema pošto će u pozivanim datotekama ostali reference na *Quarterdeck*-ove komponente. Sve u svemu ako vam QEMM lepo radi ne žurite sa prelaskom na EMM386 čak i kada ste preši na DOS 6.0!

699 [OS MSDOS, PC] Još jedna mala tajna DOS-a 6.0 ako umesto DOS=HIGH napišete DOS?=HIGH pri svakom podizanju sistema biće postavljeno pitanje DOS=HIGH [Y,N]? Znači ako što se pritiskom na F8 može inicirati izvršavanje konfiguracionih datoteka red po red uz kontrolna pitanja. Iako sa kontrolna pitanja mogu koncentrisati na određena naredbe. Osim kod DOS= upitnik može da se pojavi i pored BUFFERS INSTALL i DEVICE [Preuzelo sa Sezama, prilog **Ivan Skeri** (*skeri*)]

700 [OTXT YUZI, PC] Već duze vreme govori se o višebajtnim kodnim rasporedima koji bi konačno trebalo da reše sve problema sa nestandardnim rasporedom YU slova cirilicom konverzijama i tome slično. Evo nekoliko osnovnih informacija o nacrtu lakozvanog UCS-a, što je skraćenica za *Universal Multiple-Octat Coded Character Set*. Predlog je finalizovan tokom 1992 godine.

Svaki znak se prema ovom standardu beleži kao četiri bajta pri čemu se u oslabljenoj verziji

Latinica

Č	010C	č	010D		
Ć	0106	ć	0107		
Đ	0110	đ	0111		
Š	0160	š	0161		
Ž	017D	ž	017E		
NJ	01CA	Nj	01CB	nj	01CC
LJ	01C7	Lj	0138	lj	01C9
DŽ	01C4	Dž	01C5	dž	01C6

Latinica i ćirilica za slovanske jezike u UCS

Ćirilica

0402	Ђ
0409	Љ
040A	Њ
040B	Ћ
040F	Џ
0410-041B	АБВГДЕЖЗИ
042A-0428	КЛМНОПРСТФХЦЧШ
0430-0438	абвгдежзи
044A-0448	клмнопрстфхцчш
0452	ђ
0459	љ
045A	њ
045B	ћ
045F	џ

Listing 2

```
{ Nazavisno od kartice, program ispisuje velika C u gornjem
  levom uglu ekrana }

Var
    Adresa Word,
begin
    if Mem[0*$0448]=7 then Adresa :=$B000 else Adresa :=$B800;
    { Znak := Znak koji se upisuje + (boja pozadina Shl 4) }
    MemW[Adresa,0] :=Word('C')+($70 Shl 4),
end
```

Program za direktno upisivanje u video memoriju

zvanoj BMP (*Basic Multilingual Plane*) ili UCS-2 mogu koristiti samo dva bajta pošto se kompletan BMP sadrži u UCS-u – BMP ravan se preslikava u četvorobajtni standard tako što se ispred kodova dodaju dva nulta bajta. Već i dva bajta po znaku sasvim su dovoljna za veliki broj znakova koji osim raznih latinica i ćirilica obuhvataju hebrejski, arapski i razne kineske, japanske i koreanske ideograme.

Kontrolni kodovi standarda podudaraju se sa ASCII jer pri čemu određene kombinacije kontrolnih kodova određuju da li je raspored dvo- ili četvorobajtni. Prvih 128 znakova BMP-a su isti kao 7 bitni ASCII drugih 128 znakova BMP-a su isti kao gornjih 128 znakova ANSI (*Windows*) standarda, s tim što su kodovi 80 9F rezervisani.

Specifična slova naše latinica smeštena su na pozicijama 01xx (1 red) kao na slici 1. Na listoj slici dati su i kodovi srpske ćirilica koja je, kao što se vidi smeštena u četvrti red (04). Ruska slova sortirana su po ruskoj azbuci na pozicijama 0410-042F (velika slova) odnosno 0430-044F (mala). Makedonska slova G i K sledе posle odgovarajuceg srpskog slova, a slovo Dž je na pozicijama 0405 odnosno 0455. Staroslovenska slova su 0460-0470 [Preuzelo sa Sezama prilog **Nened Bogojević** (*nkbog*)]

701 [OTXT VENTURA, PC] Jedna veoma interesantna osobina *Venture 2.0* za GEM se izgleda izgubila u *Windows* verzijama. Nekada sa naimanje tekst mogao učitati više puta, dok se u novijim verzijama TEXT TXT u listi fajlova može neći samo jednom! Navedena osobina bila je veoma korisna ako sta na primer, na jednoj strani imali mnoštvo ramova koji sadrže isti tekst –, jedna promena u TEXT TXT odmah se odražavala na svim mestima. Novije verzije *Venture* korekitno interpretiraju poglavlja u kojima je ovaj trik korišćen, ali seme na mogu da proizvedu slična efekta. [Preuzeto sa Sezama prilog **Bojan Stojanović** (*bojansj*)]

702 [OTXT WINWORD, PC] Ako vam ne odgovara *WinWord*-ova „navika" da pri svakom startovanju otvori prozor različita veličine izaberite **Tools**/*Macro* unesite ima AUTOEXEC i kliknite na *Edit* između *Sub Main* i *End Sub* otkucajte

If AppMaximize() = 0 Then AppMaximize

Snimite ova izmena i napustite editor. Pri svakom sledećem startovanju *WinWord*-a makro će se automatski izvršiti proveriti da li je prozor već maksimiziran i ako nije proširiti ga na čitav ekran. [Preuzelo iz *PC Magazine*-a]

703 [HARDW, PC] Postoji jedna osobina memorije koja se vrlo retko navodi a koja je ipak veoma važna i objašnjava zašto isti SIM na jednoj ploči savršeno radi a na drugoj previ silne probleme. Deklarisana brzina memorija se naime odnosi na vrama prelaska memorija u aktivno stanje (*Ton*) ali nam ništa ne govori o vremenu prelaska iz aktivnog u pasivno stanja (*Toff*). Teorijski ova dva vremana bi trebalo da budu jednaka ali u praksi to gotovo nikada nije slučaj. Problam se „vuče" još iz doda diskretnih komponenti tj. pojedinačnih tranzistora tamo se uvak mogio podesiti radno opterećenja lako da sa ova dva parametra usaglase čima se dobija „idealna" prakidačka funkcija. Kod SIM-ova je sve lo malo teže – na brzim ali nekešriranim pločama dešava se da *Toff* interval bude predugačak pa sa memorija na odaziva. Po prebacivanju na kešriranu ploči memorija će odično „promisliti" pošto keš po prirodi stvan

produzava ciklus pristupa sistemskoj memoriji

Kako da proverite da li problemi koja imate počiču baš od odnosa *Ton*/*Toff*? Ako je neka memorija odbila da radi na nekešriranoj ploči (npr 386SX na 40 MHz), dodajte joj još lolko memorije pa probajte opet. Duplo više RAM a omogucice dodatno preplitanje (na SX pločama sa 4 SIM-a moguće je dvostrano preplitanje a sa 8 SIM-ova četvorostrano) čime se ciklus pristupa produžava (u datom primeru za 15-20%) [Preuzeto sa Sezama prilog **Dejan Veselinovic** (*dveselinovic*)]

704 [HARDW, PC]. I pored brojnih upozorenja da „pretvaranje" 720 K diskala u 1 44 M bušenjem dodatna rupa može ugrozrti sigumost podataka nije malo onih koji sa doste uspeha praktikuju ovu poeraciju. Problami nastupaju keda ovako udvostručenu disketu umatnemo u neki od *Notebook* računara novije generacije – podaci su potpuno ili delimično nečitljivi a disketa sa može oštatiti tako da joj ni drajv na koma je snimljena više ne može prstupiti. „Tajna" je u samoj tehnologiji izrade disketa. Iabnčki nanosi magnetnog mediuma na 720 K i 1 44 M diskelama razlikuju se u gustini kao i u razmaku medu lragovima nanosa što je uoslalom, sasvim prriodno keda sa uzma u odzir žaljani kapacitet diskala. Novija disk-jedinica znatno preciznije upravljaju glavom pa na 720 K diskalama nallaze na razne medurlragove koji Ih „zbunjuju". Ukratko oprez! [Preuzalo sa Sezama prilog **Dušan Miladinović** (*dmiladinovic*)]

705 [BASE.CLIPPER, PC] *Clipper 5.2* se prodaje u dva izdanja: amenčka verzija i verzija za „ostatak sveta". Amenčka verzija odbija da se instalira na računanma van USA i Kanade što utvrđuje proveravajući aktivnu kodnu stranicu [Preuzesto sa Sezama prilog **Pavle Paković** (*ppekovic*)]

706 [BASE.CLIPPER, PC] Konsnicima *Clipper*-a 5 svakako preporučujemo da programe prevoda navodeći opciju /W koja izaziva prijavljivanje svaka nedeklarisana promanljiva – jeste da ćete pri zasnivanju programa moriti da utrošite nešto vremena na deklarisanja svih lokalnih varijabli i (što je posebno neprijatno) polja u bazama all će sa ovo višestruko isplatiti u fazi debagovanja, pošlo će sam kompajler „prmetiti" i signalizirati razne greška u kucanju imena promenljivih koje bi inače došla do izražaja. Jak tokom izvršavanja programa. Jedna ja navolja što prelazak na /W opciju daje čitav niz lapanstvenih poruka nalik na *Warning C1003 Ambiguous reference 'GETLIST'* Ovako pravedan program radi savršeno ali je niz ovih upozoravajućih poruka prilično neprijatan – možda cete medu njima preskočiti i neko važno obaveštanja kompajlera.

„Tajna" je u tome što *Clipper* prevodi u više nivoa: postojaći *Clipper* jazik je zapravo neka vrsta prelprocesora za *Clipper* jezik nižeg nivoa a onda je taj jazik višag nivoa pretprocesor za nekakav P kod. Ukratko, GET nareoba u stvan postoji tak zbog kompatibilnosti a mlamo se lo drugačije prevodi GETLIST je jedna struktura na nivou programa uvedena baš da bi se sačuvao baram privid kompatibilnost.

Lek je jednostavan: na početak procedure treba staviti LOCAL GETLIST:={} i poruka se neće javljati! Naravno, ako proceduia poziva neku novu proceduru koja treba da „nastavi" GET onda u njoj na treba staviti LOCAL GETLIST:={} nego GETLIST pranah kao argument. Bitno je da u svakoj procedun u kojoj se koristi GETLIST bude deklarisan

707 [TEHP, PC] Kako sa podatak može upisati direkino u video memoriju pa još ne i raznim grafičkim karticama? Nije lako: video memorija se kod *Herculesa* nalazi na adresi B000:0000 dok je kod CGA/EGA/VGA na $B800:0000. Najlakši način da odredite ovu počelnu adresu je čitanje ćelije 0000:0449 – sedmica označava adresu B000 a bilo koji drugi broj adresu B800. Primer sa slike 2 ispisuje veliko C u gornjem levom uglu ekrana u odnosu na adresu Bx00:0000). Adresa Bx00:0002 odgovarala bi prvom znaku druga kolone i tako dalja – svaki drugi bajt je zapravo atribut koji opisuje boju znaka i pozadine [Preuzeto sa Sezama, prilog **Dejan Parvulov** (*dekper*)]

708 [UTIL, PC] Možala li zamisliti da će ARJ raspakovati neku PKZIP arhivu? Zvuči nemoguće ali u kompuiterakom svatu svaška može da sa desi – poslednji put smo se tako iznanadili kada smo u neku bravu graškom gurnuli pogrešan ključ (stajali su jedan do drugoga) i mimo je otključali!

Uzmita bilo koji ARJ fajl u koma poželjno je ima više datoteka i nazovite ga TEST.ARJ. Zatim izvršite PKZIP PROBA TEST.ARJ. Ako ZIP nekim slučajem uspe da sabija .ARJ makar za jedan posto (što je vrlo malo verovatno ali desi se povremenoj nadite neki drugi ARJ. Pošto je kompresija završena kažite ARJ E PROBA ZIP – ARJ će naći onu svoju arhivu „zatpanu" u ZIP i raspakovati je baz greške!

709 [MREZ, PC] Računarska mieze su i u našim uslovima sva češće a opet sa o njima relativno malo zna – termini kao što su „mreža", „server", „stanica", „terminal" i tome slično za mnoga su zbunjujući. Pokušajmo da ukratko objasnimo neka osnovna stvari

Novell mraža je najčešće mreža PC računara – u novijim varzijama u nju mogu da se uključuju Makintoši pa i VAX-ovi ali je to u praksi vrlo retko. Osnovna topologija mreže zasniva se na postojanju jednog (može ih biti i više) servera, što je obično i najjači računar u čitavom sistemu: neki 486 sa velikim diskom. Server moze da se konfiguriše kao posveceni (*dedicated*) i neposvećeni (*nondedicated*). U prvom slučaju, ava snaga servera posvećena je deobi mrežnog diska ostalim korisnicima pa bi sa mogio reći da je posvećeni server zaprav0 jedan veoma brz i moćan mrežni disk. *Nondedicated* server radi isto što i *dedicared* ali se na njemu može normalno raditi tj. pokiatati bilo koji DOS program (osim nekih specifičnih aplikacija koje zahtevaju prelazak u zaštićeni mod). Takav server će očito imati znatno manje vramena za opsluživanje mreže pa će i komunikacija po njoj biti značajno usporena. Još je neprijatnije što je stabilnost čitavog sistema ozbiljno ugrožena ako neki DOS program slartovan na serveru „zaluta" ruši se čitava mreža! Zato savetujemo korišćenje posvecenih servera na svim ozbiljnije projektovanim mrežama.

Radna stanice u mraži su takođe PC računari – njihov broj kao i posao koji će obavljati diktira i kapacitet samog servera kao i vrstu mrežnog softvera koji će se koristiti. Radna stanica u osnovi vidi server kao neki od diskova, često F – sav softver koji bi inače radio se lokalnog diska radne stanice može ukoliko je projektovan za mrežni rad bez ikakvih problema raditi i sa diske F tj. sa mrežnog diska. Osnovna (ali ne i jedina!) namena mrežnog softvera je da omogući korišćenja programa sa mrežnog diske kao i da preko njega obezbedi efikasnu razmenu podataka sa ostalim rad-

nim stanicama u mreži. Dobici su svakako pre svega ekonomski – ako pod mrežom treba da radi 20 radnih stanica, daleko je jeftinije kupiti 20 mrežnih kartica i server nego nabavljati 20 hard diskova, 20 flopi diskova i 20 disk kontrolera.

Kako se mreža sklapa? U server se ugrađuje mrežna (najčešće) *Ethernet* kartica koja je NE1000 ili NE2000 kompatibilna. NE1000 su sporije i jeftini je osmobitne kartice koje odgovaraju i XT računarima, dok NE2000 obezbeđuje znatno bržu komunikaciju uz nešto višu cenu. Zatim se disk formatira na poseban način i čitav prepušta *Novell*-u (počevši od verzije 3.0, deo diska može da se oslovi kao DOS particija, ali se na nju obično upisuju samo osnovni programi za brže podizanje sistema, na nekim ranijim verzijama *Novell*-a server se dizao sa diskete). Radna stanica se pripremа ugrađnjom mrežne kartice i startovanjem drajvera za nju sa diskete ili iz posebno pripremljenog *boot ROM*-a.

Fizičke veza između radnih stanica i servera ostvaruje se preko mrežnog kabla. To je obično koaksijalni kabl tipa RG58U ili RG58U, u novije vreme konsje se i optički kablovi. Konektor na mrežnim karticama je obično BNC tipa, kabl se ne priključuje direktno u njega, već se na karticu BNC ugradi jeden T član koji obezbeđuje povezivanje kako sa „prethodnom" tako i sa „sledećom" stanicom u lancu – na prvu i poslednju stanicu instalira se završni element (terminator) koji se sa jednog kraja uzemljava. Posloje i neki drugi tipovi kablova odnosno konektora (D-5, USA tel jack...) koji se u praksi sreću nešto ređe.

Kako se u čitavu ovu logiku uklapaju terminali? Kratko rečeno nikako – rasprostranjeno mišljenje da se na svaki jeći PC može priključiti po nekoliko „ekrana" i „tastelura" nije tačno ili bar nije tačno u DOS svetu. Mrežna kartica ne može da se utakne u terminal a sve i da može, ne bi se postigli nikakvi rezultati, jer terminal nije PC računar pa ne može raditi u PC mreži. Ukoliko PC radi pod *Unix*-om stvari se znatno menjaju i priključenje „glupih" terminala postaje sasvim jednostaven posao – umesto mrežnih kartica, konsti se običan serijski (RS 232) kabl a u glavni računer treba staviti neku multi RS-232 karticu. Iako *Unix* računer teoriјski može da opslužive veliki broj terminala, u praksi se ovaj broj retko penje iznad 4 ili 8, problem je u tome što terminali ze procesiranje konste osnovni računar koji, ma kako brz bio nije svemoguć. Se druge strane u mreži je procesiranje distribuirano po stanicama, e sa servera se samo prikupljaju podaci – tako jeći PC može da bude server za 30, 50 ili više stanica. Novije verzije *Unix*-a i *Novell*-a mogu se povezati međusobno tako da se sa jedne mreže lako prelazi na drugu, što znatno olakšava razmenu podataka u nekim većim računskim centrime [Preuzeto sa Sezama, prilog **Dušan Miledinović** (*dmladinovic*)]

710 [COMM, PC]: Sve je više konsnika kojima je elektronska pošta potrebna, ali je zbog sankcija ne mogu konsiti. Rešenje je u tome da se zakupi konsničko ime na nekom stranom sistemu, pozove taj stran sistem preko JUPAK a odatle šalje i prima pošte – jeste mnogo skupije ali funkcioniše vrlo pouzdano. Ako je „strani sistem" *CompuServe* ne morate zvati opštepoznati NUA u Sjedinjenim Državama – JUPAK Iroškoji će biti mnogo manji, eko pozovete nemačku filijalu 6234212303300 (recimo SET HOST 6234212303300 /X29 sa VAX-a). Na prompt *Host* otkucajte CIS i sve dalje ječe na uobičajeni način [Preuzeto se Sezema, prilog **Maksa Čatović** (*maxa*)]

711 [COMM, YUZI, PC]: SDK od nedavno omogućave radnim organizacijema da putem modema ostvare uvid u stanje svog računa i novčane transakcije. Čitav mehanizam je na žalost i dalje na nivou „kamenog doba" – treba najpre glasom pozvati i zamoliti službenika da ukopča modem (retke su filijale koje ostavljaju modem u *auto answer* mo-

Slika 3 BBS-ovi i informacioni sistemi kod nas i u najbližem susedstvu

BBS	Linija	Broj telefona	Radno vreme	MNP	BPS	Mreža
SEZAM – „Računari"	**15**	**011 648 422**	**Non-Stop**	**4**	**2400**	**Internet, Tanjug**
ATL	1	011 444-9459	Non-Stop	42	14400	
Durian	1	018 713-836	Non-Stop	1	2400	
ETF	8	011 324-8066	Non-Stop	4	2400	Internet,JUPAK
ELITE On.Line!	1	023 34-488	15-08	1	2400	SETNet
Euklid	1	011 620-937	Non-Stop	1	2400	
Fenix	1	011 414-312	Non-Stop	5	2400	
FON File Server	1	011 237-2393	Non-Stop	5	9600	Internet
	1	011 237 2392	Non-Stop	5	9600	
Force Aracle	1	011 507-986	20-06	1	2400	
	1	011 505-754		1	2400	
Helias	1	011 283 1367	22-08	1	2400	
Hehbiton	2	015 82-455	Non-Stop	42	2400	Internet
Huter	1	011 788 189	22-07	1	2400	SETNet
Imsi	1	011 141 992	Non-Stop	42	14400	
INFOx	1	025 28-228	Non-Stop	1	2400	
Kuchewo	1	012 52 109	22-08	1	2400	SETNet
Milimays	1	011 133-610	23-05	42	2400	
Narodna Bibl. Srbije	1	011 458-846	Non-Stop	5	2400	JUPAK
Oreska	1	031 28 276	23-08	42	2400	SETNet
Politika	1	011 322-91 48	0-24	42	14400	
Prometheus	1	024 51 792	Non Stop	42	14400	SETNet
Radio Pingvin	1	011 606-928	Non Stop	1	9600	
Rhinoceros	1	021 350-751	Non Stop	5	2400	
	1	021 350-941	Non Stop	5	2400	
	1	021 350-926	Non Stop	1	2400	
	1	021 54-857	21-08	1	2400	
Rimtutituki	1	011 872-147	18-12	5	2400	
Savage	1	011 493-982	23-06	1	2400	
Sledge HAMMER	1	011 183-452	22-07	42	2400	
Snake Dance	1	011 686-737	18-05	42	2400	SETNet
Spice	1	011 873-739	23-06	1	2400	
Tangram	1	018 21-661	Non-Stop	1	2400	
TIGAR Pirot	1	010 37 277	Non Stop	5	2400	
Velkom	1	011 344-263	Non-Stop	5	2400	FIDO
	1	011 628-468		1	2400	
Wizard	1	011 488-1467	22-08	5	2400	
ZaMir-BG	1	011 632-566	Non-Stop	42	19200	Internet

du) i onde se uspostavlje modemska veza posredstvom posebnog komunikacionog programa koji SDK „dell" zainteresovanima za ovu uslugu (program je inače strane proizvodnje i veliko je prtanje da li je PD/SW).

Pojavljuju se veliki problemi sa kompatibilnošću modema – praksa je pokazala da treba nabaviti MNP5 modem i onda ne njemu iskjučiti MNP emulaciju, zvuči šašavo, ali je to jedina kombinacija koja radi! Mogu se pojaviti i problemi se identične cijom koju terminal šalje *host*-u što bi SDK trebalo da podesi pri instalaciji čitave opreme [Preuzeto sa Sezame, prilog **Zoran Životić** (*zzivotic*)]

712 [COMM, YUZI]: Na slici 3 je spisak svih domaćih BBS-ova i informacionih sistema. Osim ubičajenog broja telefona radnog vremena i parametara veze navedene su i neke specijalne usluge pojedinih sistema – *Internet* obuhvata slanje i prijem elektronske pošte iz sveta (za sada je obim ove usluge veoma ograničen zbog sankcija UN), FIDO je međunarodna mreža BBS-ova, a SejNET domaća mreža 7-8 manjih BBS ove iz Srbije i Makedonije koji radovno razmenjuju privatnu poštu i konferencije. Prijatno zvanje!

713 [IGRE.PRINCE, PC]: Posle popularnosti arkadne avanjure *Prince* koja se tokom 1991. i 1992. igrala na „svakom" PC ju u Jugoslaviji ni *Prince 2* nije ostao bez brojnih pristalica. A kada govorimo o igri svakako govorimo i o „varenju" u njoj. Ovoga puta tako našlo su predvideli i sami autori – dovoljno je da startujete program sa PRINCE MAKINIT i biće opremljeni nekim „natprirodnim" moćima – sa Alt N prelazite na sledeći ekran, pritiskom na K ubijate sve protivnike na ekranu sa Alt T dodajete sebi života, a se Shift S „izlazite iz kože" i postajete senke neviđjiva za sve protivnike. Dodatne funkcije dodaljene su i tasterima F2, F3, F6, Shift K, Shift W, Shift R, Shift I, Shift B, Shift T i Alt D, ali ni jedne od njih nije od naročitog značaja za igranje – verovatno su ih konstili programeri u fazi testiranja programa. Kasnije izdanja *Prince*-a 2 (koja nismo videli u našim krajevima ali ipak da pomenemo) se u ovaj *cheat mode* dovode ne sa PRINCE MAKINIT nego sa PRINCE YIPPEEYAHOO.

Sve do varanje vam ipak neće pomoći da završite igru pošto ne poslednjem (i najelektrnijem) ekranu ubijanje sa K „ne pali". Dođite na ledeni deo i pobijte sve svoje kopije osim jedne koja će sama nestati. Popnite se gore i na zuloj mermernoj osnovi okrećite brzo levo-desno više puta. Posle kraćeg vremena primetićete de svetiluće i gubite život – ali naslavite da se u mestu okrecete dok vaš duh ne izađe iz tele koje u istom trenutku ostaje mrtvo. Onda kao duh podite udesno i na sledećem ekranu videćete čarobnjaka koji se „meskirao" u ves. Podite bez žurbe ka njemu i on će sići u dno ekrana. I dalje ga pratite i čarobnjak će skočiti ulevo. Zaletite se i skočite za njim držeći Shift da biste se uhvatili za led. Popnite se i primetićete čarobnjaka dve reda iznad ves. Dođite polako do srednjeg reda, popnite se i u tom trenutku čarobnjak će potrčati. Jurnite i vi udesno sli umesto običnog kretanja strelicom prijsnite per puta i strelicu ne gore kako biste u trku skočili – tako idete brže i „uletećete" pravo u čarobnjake koji će iz nepoznatih razloga pri sudaru čučnuti i ostati u čučećem stavu. Odatle da se izmaknete par koraka ulevo okrenete ka njemu i pritisnete Ctrl da biste bacili kletvu. To je sve – čarobnjak je mrtav princeza probuđene igre završene – ipak obratite pažnju na čarobnu kuglu koja se pojavljuje na samom kraju i ružno lice koje u nju gleda – možemo da očekujemo i *Prince*-a 3! [Preuzeto sa Sezame, kompilacija brojnih priloga više korisnika]

Priloge za ovu rubriku šaljite na adresu „Računari" (za „Bajtove lične prirode"). Bulevar vojvode Mišića 17, Beograd ili preko Sezama u vidu lične poruke uredniku (*meli write dejan*).

Novo u izdanju
Mikro knjige

AutoCAD 11

Prevod čuvene SYBEX-ove knjige
Mastering AutoCAD, Version 11

Kompletan vodič kroz AutoCAD Sve što ste želeli da znate o AutoCAD u konačno na jednom mestu početak rada sa AutoCAD-om, kreiranje biblioteka simbola i oblika, rad sa tekstom i ostalim oznakama, povezivanje crteža sa bazama delova, materijala i drugih elemenata, izračunavanje dimenzija i razmera, unošenje skica, korišćenje štampača i plotera, AutoLISP programi, rad sa trodimenzionalnim crtežima, povezivanje sa stonim izdavaštvom, i još mnogo toga

784 strane, 571 slika, format 16,5 × 23,5 cm

Iskoristite priliku i nabavite ovo vredno delo direktno od izdavača

❖

Najveći izbor domaće i strane kompjuterske literature Javite se za besplatan katalog naših izdanja Najpovoljniji i najsigurniji način nabavke – direktno od Mikro knjige

Mikro knjiga
Petra Martinovića 6, 11030 Beograd
tel 542 516 i 542 619, fax 542 516

HARDVER

Blasteri slike i tona

Novosti iz kreativne laboratorije

„CreativeLabs" firma čuvena po zvučnoj ploči *SoundBlaster* sada predstavlja čitav niz novih i najav ljuje još neke proizvode Između detalog tu je i nova verzija digitalne kartice za hvatanje video slike *VideoSpigot for Windows* Nalazi se u paketu sa programi me *Cinepak* (za kompresiju i dekompresiju) i *Microsoft Video for Windows VideoSpigot* nudi brzinu od 30 ekrane u sekundi e preporučena cena je 399 do lara

Ističe se da je „Creativa" prva kompanija koja pri menjuje i prodaja *Cinepak codec* tehnologiju preuze tu od proizvođača perfemih grafičkih uređaje Super- MacTechnology *Cinepak* je softverski video kompre sor/dekompresor koji postiže stepen kompresije od 20 1 praktično bez smanjenog kvaliteta slike u odno su na originalne nakomprimovane 24-bitne AVI datoteke

U pripremi je i veliki broj video-konferencijskih proizvoda za *Windows* koji primenjuju „ShareVision" tehnologiju koje je otkupljena u junu mesecu *ShareView*i *ShareView Plus* rade preko standardne telefon ske žice i omogućavaju korisnicima istovremen rad na letom dokumentu u stvarnom vremenu bez obzira da li imaju instaliran isti softver *ShareView Plus* obezbe đuje živi kolor video oba konsnika u stvarnom vreme nu

ShareView nudi i vlastitu video-kompresiju čiji stepen dostiže 350 t

SoundBlastar DigitalEdge CD je multimedijalna dogradnja za PC koje sadrži *SoundBlaster 16* sa unapređenom obradom signala I CD-ROM drajv sa udvostručenom brzinom Kažu da prevazilazi mogućnosti MPC Level 2 specifikacije Nudi hardversku kompresiju u stvarnom vremenu dok interni je CD ROM drajv Photo-CD kompatibilan I postiže brzinu pristupa od 350 milisekundi

U softverskom paketu se nalaze *VoiceAssist* pro gram za prepoznavanje glasa softver za multimedije programi za obradu teksta i upravljanje bazama poda taka alatke za razvoj le mikrofon i zvučnici Veliko prodajna cena za *Sound Blaster DigitalEdge CD* izno si 999 USD

„Creative" takođe najavljuje da će se od sredine jula *VoiceAssist* prodavati kao samostalan *Windows* softver za većinu zvučnih kartica za PC uključujući i 8- I 16-bitne SoundBlaster karte Ovaj program auto matski prepoznaje i aktivira sve meni komande u *Win dows* aplikacijama i omogućava kreiranje korisničkog rečnika od 30 000 reči Kompatibilan je sa većinom zvučnih ploča koje imaju *Windows 3 1* drajvers i mo gućnost ulaza preko mikrofona I dalje će se prodavati zajedno sa Sound Blaster karticama dok će kao sa mostalan proizvod sa mikrofonom, koštati 99 USD

Konsne adresa Creative Labs Inc tel 408-428- 6600

PowerPC

Motorola u protivnapadu: Brže i jeftinije od Pentiuma

Motorola je obelodanila planove za reklamnu kampanju kojom želi da ubedi proizvođače i konsnika personalnih računara da su njeni mikroprocesori po djednako dobri pa čak i bolji od Intelovih Agresivni reklamni pohod Motorole treba da promoviše nove RISC PowerPC mikroprocesore Tvrdi se da su *Po werPc* čipovi brži manji i štedljiviji od Intelovog netom izašlog Pentiuma Snažana adut je i razlika u ceni – tvrdi se da će *PowerPC* sa cenom od 450 dolara, biti upola jeftiniji od Intelovog Pentiuma za koji se predvi đa cena od 965 dolara

U međuvremenu, analitičari pokušavaju da pred vide kakva budućnost stoji pred Intelom Mnogi veruju da RISC procesori predstavljaju ozbiljnu i trajnu pret nju sa kojom će Intel morati najozbiljnija da se suoči

Kažu da je najavljena kampanja Motorole unele narvozu u Intelove redove jer nagoveštava mogućnost ljute bitke između dva mikroprocesora Loša vest za Intel ali dobra za konsnike jer takva konkurencija može doneti samo niže cene

#define MAKRO

Već je nebrojeno puta rečeno da je jedno od lepih osobina C-a njegove prenosivost – napišete program koji radi pod DOS-om zatim prenesete izvorni kod na neku UNIX mašinu pravdeta program i sve radi i zaista ako ste imali prilike da ovako nešto probate mogli ste se uveriti da tvrdnja o prenosivosti C-a nije bez osnova Ipak već površna analize programa koji se ovako lako može proštastiti od jednog okruženja do drugog će otkriti da iza priče o prenosivosti stoji i jedne mnogo-komplikovanije priče o naporima programera koji takav program pišu Ispostaviće se zapisvo da se u jednom programu nalaze pojmovi odvojeni delovi koda posvećeni pojedinim okruženjima – kao da ste napisali nekoliko verzija programa

Zahvaljujući mogućnostima pretprocesora ovaj posao lako napotan može na kraju razuiovati sa svim čtkim programom Preglednost i jednostavnost modifikacija kode koje se može postići propisnim korišćenjem pretprocesora je tolika da je prilično čudno da se ova faza prevođenja na konsti i kod svih ostalih kompajlera Nema čak n kakve prepreke da se recimo i pretprocesor nekog C kompajlera iskoristi za pretnodnu obradu pascal programa

Osnovna pretprocesorska direktiva je **#define**

#define REČ tekst_kojim_se_zamenjuje
#define REČ(lista parametara) tekst_kojim_se_zamenjuje

U prvom slučaju koji u praksi obično izgleda

#define MAX_ROWS 80

radi se o deklarativnoj konsteni U drugom slučaju kada konstimo argumenta

#define KVADRAT(x) (x*x)

radi se o makrou U oba slučaja princip i ada pretkompajlera je isti Da biste shvatili kako pretprocesor obrađuje **#define** d rektivu poslužićemo se jednim potpuno neprogramskim primerom Napišete sledeći tekst

#define PERA(x) perine x

Ovo je tekst u kome će makro "PERA(ime) biti zamenjen se PERA(time)

Ako sada ovaj tekst „prevedete" kompajlerom uz opciju EP (samo pretprocesiranje i ispis rezultata na ekran) dobićete

Ovo je tekst u kome će makro "PERA(ime) biti zamenjen sa perine ime.

Dakle pretprocesor radi se tekstom ne znajući skoro ništa o programima Skoro ništa zato što ipak podrazumeva neka konvencija bitne za pisanje programa a jedne od njih se vidi i ovde – tekst unutar znakova navoda smatra se nedodirljivim jer se u programime na ovaj način definiše sadržaj stringa Slučajno poduđarsanje neke reči se makroom mogio bi nepravti zaburu Druga konvencije je poduđarsanja znakove koji se smatraju separatorima reči – ne možemo definiseti makro koji će na primer „MOJE IME" zamenti nekim drugim tekstom jer naziv makroa prestaje kod prvog takozvanog **white-space** znake (razmak ili tabulator) Slično je i sa parametrima makroa – nije moguće napisati

#define TEST(x,y) xy

i očekivati da će **TEST(a,b)** biti zamenjano se **ab** jer parametri moraju biti jasno razdvojeni

Poslednje makro zahteva da se ceo tekst kojim se zamenjuje nalazi u jednoj linji Produženje sadrža ja na više linija je moguće ali na isti način kao i kod višelinijskih strigova – dodavanjem znaka \ na kraju reda

#define TEST(x,y) for(x=0;x<y,++x)
niz[x] = x,

I konačno konvencije ja da se svi makroi i konstanine pišu velikim slovima Ovo je isključivo konvencija – pretprocesor ne pravi ograničenja ali ovde ne treba biti originalan Ako očekujete da će vaš program analizirati drug programeri jeden od sigurnih načina da ih iznervirate je da b rale originalno napisana imena (mala slova ili mešano) kako bi umeslo da odmah pronađu sadržaj makroa u H zaglavlju shvate o čemu se radi bezuspešno tražili funkciju po modulima programa

O nekim sitnim trikovima i operatorima koji dodatno povećavaju mogućnosti makroa bilo je reči u jednom od prošlih tekstova iz ove serije

Svakako najvažnije pitanje je – kada upotrebljavati konstante i makroe? Odgovor naravno ne može biti dat u obliku konečnog skupa pravila ali se neke praporuke pak mogu dati Najčešća skoro obavezna primena je definisanje konstanti – sve dimenzionisa nja nizova granične vrednosti i slično su prava mesto za primenu Malo je nizove koji kao tad u programu neće bti indeksirani do svoje granice i ako ona nije definisana kao konstanta biće potrebno prilično truda da se nakon promene dimenzija niza pronađu sva ta mesta i na njima upiše nove vrednost

Pn konščenju deklarativnih konstanti ipak ne treba pratenvati U programima poneksao možete sresti čak i ovakvu situaciju

if (line[i] == REW_LINE)

—

Prva pomsao je da je konstanta NEW_LINE neki specijalni marker koji je autor usvojio kao terminalor Onda pogledate zaglavlje i tamo pronađete

#define REW_LINE '\n'

Naravno na treba isključiti mogućnost da je autor zamislio da terminator može biti i neki drugi znak pa je na ovaj način ostavio otvorena vrata za laku izmenu ali ipak je češći slučaj da se radi o „preteranoj čistoći" pisanje

Drugi slučaj doduše nešto delikatnije prirode nepažljive upotrebe konstanti ilustruje sledeći primer

#define FATAL_ERROR "Fatalna greska – prekid rada"

fprintf(stderr,FATAL_ERROR);

Principijelno se ovoma ne može nešto posebno zameriti ali se praktične lečke gledanja ime jednu velikiu manu Ako se neme ispis poruke pojavljuje na mnogo mesta u programu na ovaj način smo samo umnožili pojavu stringa u kome je tekst poruke jer ga kompajler nakon što preiprocesor izvrši zamenu sva ki put vidi kao novi string Doduše neki kompajlen imaju opciju za spajanje duplih stringova u jedan ali je ovo reda i ne uvek primenjiva mogućnost U ovim slučajevima daleko je bolje konstih

char *FATAL_ERROR = "Fatalna greska – prekid rada";

Pravila za upotrebu makroa je teže dati Jedna od najčešćih situacija je obezbeđivanje kompatibilnosti funkcija različnih verzija kompajlera Na primer ako želimo pozicioniranja kursora na ekranu što je nestandardne funkcija onda možemo uvesti makro **GOTO(x,y)** koji se potom može definisati kao

#if defined(MICROSOFT)
#define GOTO(x,y) _settext(position(x,y)
#elseif defined(BORLAND)
#define GDTO(x,y) gotoxy(x,y)
#else // ANSI DRAJVER
#define GOTO(x,y) printf(\033[" #x ";" #y "H),
#endif

U prve dva slučaja popularnih kompajlera se za pozicioniranje kursora koriste odgovarajuće bibliotečke funkcije dok u ostalim slučajavima pretpostavlja mo da nemamo odgovarajuću funkciju pa koristimo ANSI drajver ispisam odgovarajuće kontrolne sekvence

Druge takođe česte upotrebe je grupisanje više informacija jednim makroom Ako neka funkcija traži dva parametra pointei na niz i dimenziju niza

funkcija(t, sizeof(t)/sizeof(t[0])),

onda je korisno napraviti makro

#define RIZPAR(x) x, sizeof(x)/sizeof(x[0])

i potom pisati

funkcija(RIZPAR(t)),

Nje samo skraćeno pisanje razlog za ovaj makro, već sva informacije koje su čvrsio vezane ze sam niz ne neki način apstrakujemo pa se sventualne promene nnoge što je funkcija potrebno lako izvoditi Ako us lanovimo da je bolje da pnnesemo pointei na niz i dimenziju elemenia (umesio broja elemenata) dovolj no je samo izmeniti makro

#define RIZPAR(x) x, sizeof(x[0])

I ostatak programe ne treba dirati

Skraćeno pisanje je naravno dovoljno čest razlog da poseqnemo za makroom Česta ponavljanje nekog dela kode ima smisle izdvojiti kao makro na primer

#define TERMINATE(s){ *s = '' < 32 || *s > 127)
if (TERMINATE(s))
break,

U ovakvom primeru se može naći i dodatnih razloga za opravdanost uporebe makroa – konstante koje tražimo u nizu se lako mogu promenti može se ubaciti dodatni uslov i slično

Konačno makroima se često ostavlja i sloboda u odluci da li neki deo koda izdvojiti kao funciju ili pisati *inline* Na primer

#define REPLACE(s,x,y) {
while (*s)
if (*s == x) *s++ = y, else s++, }

Ako ustanovimo da ipak ima smisle zamenu jednog znaka drugim u celom stringu napisati kao posebnu funkciju dovoljna će biti samo izmena makroa.

#define REPLACE(s,x,y) replace(s,x,y)

pn čemu je **replace** novo napisana funkcija

I kod upotrebe makroa ne treba pretenvati Ponekad se može sreti program u kome su čitavi blokovi koda u stvari složeni makroi čime se kod nepotrebno gomila bez ikakvog dobitka u čitkosti Ne treba opet otići ni u drugu krajnost pa potpuno odustati od makroa kojima definišete blok koda Ovo je ponekad dobar način za ručno dotarivanje optimizacije – *inline* kod kojim se makro zameni može neznatno povećati količinu generisanog mašinskog koda ali i brzinu izvršavanja programa u odnosu na varijantu sa pisanjem funkcija

Kod konstruisanja makroa potrebno je nešto dodatne pažnje u odnosu na direktno pisanje koda u samom programu **Prve pravila** je da ne štedite ne zagrađama Ako makro

#define ADD(a,b) a+b

upotrebite u izrazu

c = 10*ADD(x,y).

izraz se nakon zamene računa kao „10*a + b" što verovatno nije ono što se želeli Dakle izvorno je

#define ADD(a,b) (a+b)

Drugo pravilo je da stalno imate na umu da se radi o *inline* kodu ako makro izgleda „ko funkcija" Ako pažljivo pogledate sadržaj prethodnog primera RE PLACE makroa primetićete da se u njemu vrši inkrementranje vrednosti string pomtera. To konkretno znači da nakon

REPLACE(str, '\', '/'),

pointei **str** pokazuje na kraj stringa što ne bi bio slučaj da je u pitanju pozv funkcija jer bi se vrednost **str** pointeia prenela kao argument i ona bila menjana a ne originalni pointai Takođe kao parametai ne možete proneti ne pomer

REPLACE(str+x, '\', '/'),

jer će nakon zamene tekstа to prerasti u

while (*str+x) {
if (*str+x == '\') *str+x+ = '/', else str+x++, }

što više nema veze sa sintaksom C-a

Već klasično upozorenje se odnosi ne sličan problem – ako ranije pomenut primer TERMINATE makroa u programu iskoristite na sledeći način

if (TERMINATE(a++))
break,

previte latalnu grešku jer će nekon makro ekspanzije logički izraz glasiti

if (*s++ == < 32 || *s- + > 127)

što će zbog višestrukog inkrementiranja koje se očigledno pojavilo napraviti veliki problem programu

Iz svega do sada rečenog generalno bi se mogao izvući zaključak da makroa i konstante treba maksimalno koristi ali samo do onog trenutka do kog su još uvek jednostavni jasn i lako primenjivi u programu bez ikakvog razmišljanja o njihovom stvarnom sadržaju i ne kraju jedan mak test.

#define REKISTRING "abcd"

char *s,
if (*s == *LIMITS)
{
if (*s == *(LIMITS+1))

Kako će izgledati logički izrazi nakon zamene makroa i da li će kompajleri „progutati" tako nešto?

List operator

Svi veći priručnici koji se isporučuju uz nove programe izgleda da imaju suprotan efekat od onoga koji bi želeli njihovi autori umesto da korisnici sve više znaju o dotičnim programima sve je više mračnih kutaka koji ostaju nepoznati. Za primer jeste li ikada videli ovakav *Clipper* program

```
PROC Main
  (1, 2, 3)
RETURN
```

Verovali ili ne, ovo je sintaksno sasvim ispravno i pre vođenje će program prevesti bez problema. Doduše, program ne radi ništa korisno ali ilustruje jednu malo poznatu konstrukciju pozamašnju iz C-a: list operator. Ta konstrukcija u stvari predstavlja ništa drugo nego niz izraza razdvojenih zarezima. Takva lista je i sama izraz čija je vrednost tog izraza je vrednost poslednjeg elementa. Na primer, u naredbi

```
d := (y+2, z, "ABC")
```

promenljiva **D** će dobiti vrednost **"ABC"**.

Ovaj operator možete upotrebiti prilikom računanja u izrazima, ali s obzirom na kvalitet *Clipper*-ovog optimizatora koda, time se jedino dobija nečitljivi kod. Prava primena je u kodnim blokovima. Nažalost, tamo često zatreba neka kraća konstrukcija i tada je mnogo preglednije pisati to u okviru kodnog bloka nego u njemu pozivati posebnu funkciju. Problem nastaje kada je potrebno izvršiti neko grananje. Tada se obično koristi uslovni izraz

```
IIF(x == y                    ,
   QOut("="),                 ,
   QOut("<>")                 ,
)
```

Međutim, šta ako je potrebno u jednom od slučajeva izvršiti ne jednu već više naredbi izraza? Naravno, te da se može napisati funkcija koja sadrži te naredbe ali ipak je mnogo bolje rešenje list operatori

```
IIF(x == y                    ,
   (Beep(2), QOut("=")),      ,
   QOut("<>")                 ,
)
```

ON ERROR

Sigurno se sećate *BASIC*-a i njegove ponekad veoma zgodne naredbe **GOTO**. Neki bolji interpreteri su imali i naredbu **ON ERROR** koja je omogućavala da se u slučaju nepravilne greške uradi proizvoljna akcija. Obično je ta akcija bila upravo **GOTO** a cilj je bio početak glavnog menija. Teško bi korisnik posle nastale greške video sama kratku pauzu zbog ispisivanja glavnog menija. *Clipper* ima mnogo finiji sistem za kontrolu grešaka od *BASIC*-a, ali nema nešto što mnogi ponekad poželjuju. **GOTO** naredbu za skok na proizvoljnu naredbu. Ipak, ove naredbe se ugla vnom može simulirati konstrukcijom **BEGIN/END SEQUENCE**. O ovoj konstrukciji je bilo reči u jednom od prethodnih brojeva „Računara". Podsetimo se: svaka naredba **BREAK** između **BEGIN SEQUENCE** i **END SEQUENCE** zazvaće skok na prvu naredbu iza **END SEQUENCE**.

O tome kako se mogu kontrolisati greške u programime bilo je takođe reči. Ukratko: treba između procedure koju sistem za obradu grešaka poziva kada se dogodi greška. U toj proceduri se može uraditi proizvoljna akcija a uobičajeno je da se korisniku obaveštenje da je nastala greška. Na kraju se prekida program naredbom **QUIT**. Međutim, dosta programera ne želi da program „skoči" već da se izvrši automatsko „restartovanje" programa. Iako korisnik ne mora da se uključuje u za njega neprijatne *DOS* komande.

Prvo se mora odrediti mesto na koje će se izvršiti skok u slučaju greške. Idealan kandidat je početak glavnog menija: svaki veći program ima u nekoj formi glavnu raskrsnicu na kojoj korisnik bira dalju obradu. Sve procedure koje se pozivaju iz okvira u „raskrsnici" treba obuhvatiti **BEGIN/END** blokom

```
WHILE i <> 0
   CLS
   i := Menu()
   BEGIN SEQUENCE
   DO CASE
   CASE i == 1 , Obrada1()
   CASE i == 2 , Obrada2()

   END CASE
   END SEQUENCE
END DO
```

Sada će svaka naredba **BREAK** u bilo kojoj proceduri **Obrada()** izazvati skok na kraj **WHILE** ciklusa, što znači da će se program ponovo naći u glavnom meniju.

Sledeći korak je prepravka sistema za obradu grešake o tome je bilo reči ranije. Bilo da prepravljate **ERRORSYS.PRG** ili ste napisali svoju funkciju, sada je jedno važno da umesto sa **QUIT** obradu greške završite naredbom **BREAK**. Iako će se kontrola vratiti na glavni meni. Treba voditi računa o tipu greške i o ponovnoj normalizaciji sistema, pre naredbe **BREAK** treba obavezno pozatvarati otvorene datoteke i sl.

Makroi u kodnim blokovima

Kodni blokovi mogu poslužiti kao zamena za makroe uvek kada je izraz koji treba interpretirati poznat za vreme prevođenja programa. Kad god je to moguće bolje je koristiti kodne blokove – oni se brže izvršavaju, kod je jasniji i sl. Međutim, u nekim slučajevima se mora upotrebiti makro, ako izraz koji treba izračunati nije poznat za vreme pisanja programa, onda nema druge nego upotrebiti makro operator. Međutim, i tu ima razlika: ako je potrebno samo jednom upotrebiti makro operator, onda je sve u redu, ali ako se to ponavlja više puta (u nekom ciklusu, na primer), svaki put će se iznova vršiti analiza i prevođenje izraza što se i te kako može odraziti na brzinu izvršavanja programa. Teda je najbolje koristiti kombinaciju kodnih blokova i makroa. Kodni blok se definiše ovako

```
bBlk := {|| &cExpr }
```

Promenljiva **bBlk** dobija kao vrednost kodni blok koji se kasnije tek za vreme izvršavanja programa, što znači da nije potrebno znati izraz za vreme prevođenja programa. Kreirani kodni blok je u stvari „prevod" zadatog izraza, pa će se izvršavati daleko brže nego **&cExpr**.

Nekad je kombinacija kodni blok – makro neophodna. Na primer, pri kreiranju tabelarnog prikaza za neku matricu treba zadati kodni blok koji izračunava tekući element matrice. Ako su i i j globalne promenljive koje čuvaju tekuću poziciju kursora u nizu tada bi tekući element bio

```
{|| a[i,j]}
```

Međutim, pošto jedan kodni blok treba definisati za svaku kolonu pa bi to kodni blokovi izgledali ovako

```
{|| a[1,1]}  {|| a[1,2]}  {|| a[1,3]} ...
```

Ali ako pokušate da sledeću naredbama zadate kolone za prikaz

```
FOR j := 1 TO 10
   TB:addColumn(TBColumnNew("" , a[i,j]) )
NEXT
```

nećete dobiti ono što ste želeli: sada će sve kolone biti opisane istim kodnim blokom pa će se uvek prikazivati jedna kolona (u zavisnosti od vrednosti promenljive j). U stvari potrebno je da se "i" računa za vreme izvršavanja a "j" za vreme definisanja kodnog bloka. To se postiže na sledeći način

```
FOR i := 1 TO 10
   cTmp := "a[i," + Str(j) + "]"
   TB:addColumn(TBColumnNew("",{|| &cTmp }))
NEXT
```

Primetite da se makro operator izvršava za vreme definisanja kodnog bloka. To se naziva i kao **izvršavanje makro operatora**. Ponekad je potrebno da se izvršavanje odloži i da se makro izvrši kada se računa i kodni blok. Na primer, ako je potrebno definisati kodni blok koji dodeljuje neku vrednost promenljivoj zadatog naziva sledeća neće raditi

```
x := 5
bAssign := {|x,y| &x := y}
Eval (bAssign, "x", 123)
```

Naime, pošto se makro izvršava za vreme definisanja kodnog bloka, rezultujući kodni blok biće određen tada pa će funkcija **Eval** pokušati da izvrši kodni blok `{|x,y| 1 := y }` što je naravno greška. Izvršavanje makro operatora se može odložiti primenom makro izraza jednostavno treba zatvoriti zaljeni string u zagrade. Ispravan primer glasi

```
x := 1 // Nevazno
bAssign := {|x,y| &(x) := y}
Eval (bAssign, "x", 123)
```

Sada će promenljiva **X** dobiti vrednost 123.

ZANIMLJIVOSTI

Zaštita podataka

Vodite računa na vreme, posle će biti kasno

Na otvaranju izložbe PC Expo poslednje nedelje juna, „3M Data Storage Products" je promovisao nagrade za zaštitu podataka za ovu godinu ustanovljene u saradnji sa desetinom korisničkih grupa koje su odlučile da daju svoj doprinos sigurnosti podataka. Kažu da godišnji gubici koje američke organizacije trpe od neadekvatne zaštite podataka iznose 4 milijarde dolara.

Ovaj program 3M-a predviđa nagrade za pojedince koji su kreirali izuzetne sisteme za *backup* i zaštitu podataka na PC računarima poslovnih korisnika. Ideje je proizišle iz studije „Intelliquest"-a (za 1992 godinu) koje pokazuju da dve trećine korisnika u američkom biznisu ili ne primenjuju formalnu zaštitu podataka, ili čak nisu ni svesni da takvo nešto uopšte postoji. Značajne gubitke podataka je imala svaka treća ili četvrta ustanova dok polovine svih poslovnih korisnika očekuje gubitke kad tad. Pri tome samo 18% manjih firmi vodi računa o redovnom i pouzdanom bekapu dok kod velikih preduzeća taj broj iznosi oko 40%.

„Nekoliko novih trendova – uspon sistema distribuirenih informacija, proširenje skladišnih kapaciteta personalnih računara kao i proširan prostup sofisticiranim aplikacijama i velikim količinama informacija – povećali su vrednost ali i ranjivost podataka na PC sistemima." kaže predsatvnik kompanije 3M „Oni koji su se potrudili da povežu bekap i zaštitu podataka i ukupno poslovno zdravlje svojih preduzeća zaslužuju javno priznanje."

U žiriju će biti novinari predstavnici zainteresovanih američkih korporacija i vođe korisničkih grupa, a nagrade će se dodeliti na osnovu pet kriterija: originalnost komunikacija motivacija i kreativno mišljenje. Pobednici njihove organizacije kao i oni koji su ih nominovali će dobiti priznanja na jesenjem Comdex-u.

OBNAVLJANJE TORNERA I RIBONA

„B O L E K O"

Tel 011/3225-952

VENTURA

Beograd Yugoslavia 27 marta 70 Tel/fax 38(0)11-346-771
POVRATNE KARTE SA TRANSFEROM DO AERODROMA

MOSKVA	245 $	CHICAGO	628
LARNACA	250	L.A.	776
RIM	262	RIO DE JANEIRO	1100
MILANO	226	TOKYO	1100
PARIS	269	SAO PAOLO	1100
AMSTERDAM	280	BANGKOK	1100
TRST	240	CARACAS	1100
SINGAPORE	860	HONG KONG	1100
TORONTO	548	LONDON	366
JOHANESBURG	750	FRANKFURT	298
SIDNEY/		HANBURG	378
/MELBOURN	1340	CIRIH	232
NEW YORK	600	COPENHAGEN	295
		PEKING	765

PLUS

SVA PUTOVANJA U ZEMLJI I U SVETU

ISKORISTITE MOGUĆNOSTI DINARSKOG PLAĆANJA ILI SA STARE DEVIZNE ŠTEDNJE

Logovanje

Kada korisnik ukuca svoje korisničko ime (*user name*) i odgovarajuću šifru (*password*) program Jogin* u datoteci **/etc/pasawd** u kojoj se nalaze osnovni podaci o svim korisnicima na sistemu pronalazi zapis se osnovnim podacima za tog korsnika Zapis za korisnika *paya* bi mogao da izgleda ovako

paya*:200 50 Pavle Pekovic, FON./usr1/paya:/usr/bin/bash

Kao što se vidi podaci su međusobno razdvojeni dvotačkama Prvi ja korsničko ime dok naredna tri u ovom tranutku nisu zanimljiva Peto polje najčešće sadrži ime i prezime korsnika i aventuarno zanimanja i naziv firme u kojoj radi Šesto i sedmo polje su osnovni direktorijum korsnike (*home directory*) kao i osnovni program (*shell*)

Po uspešnom prijavljivanju na sistem startuje se osnovni program (*shell*) pročitan iz **/etc/pasewd** datoteka Iako se kao *shell* može staviti bilo koji program najčešće se konsta tzv komandni proceson Zbog toga se pojem *shell* ako se drugačije ne nagla si..konsi za komandna procesore Da bi MS-DOS korsnicima bio jasniji pojam *shell*-a recimo da je *shell* na Unix-u isto što i **command.com** ili **4dos.com** na MS-DOS-u Pošto su *shell*-ovi, njihove mogućnosti i međusobna razlika široka lama, ostavimo je za neki od sledeći brojeva „Računara"

Sam *shell* startuje komande iz **/etc/profile** datoteke u kojoj se najčešće definišu neka osnovna setovanja za sve korisnika prikazuje se tzv poruka dana (*message of the day*) iz datoteke **/etc/motd**, provarava da li ima nove pošta za korsnika novih ili ws poruka i sl Sve ovo je moguće izmeniti u **/etc/profile** stavih neke druge komande ali je jo pnvilegija administratora sistema, što je i logično, jer se **/etc/profila** izvršava za sve korsnika Nakon toga se izvršava korsnikov profila (datoteka **profile** u osnovnom direktorijumu) u kojem korsnik moze sam definisali svoje komande setovanja i sl U slučaju da korsnik koristi *csh* (*C shell*) umesto **/etc/profile** i **./profile** se izvršavaju **/etc/cshrc** i **./cshrc** Porad njih kad je **csh** u pitanju izvršava se i **./login** datoteka odmah nakon **./cshrc** Da napomenem da je " " oznaka za osnovni direktorijum korsnika Tipičan **profile** je pnkazan na slici 1

SHELL, HOME PATH PS1 i TERM su jedne od najvažnijih *environment* promenljivih koje predstavljaju *shell* osnovni direktorijum *path* izgled prompta i tip terminala respektivno Komandom **axport** se ova promenljive definišu i za sekundarne *shell*-ove Zanimljiva je promenljive PS1 u kojoj definišemo izgled prompta. Moguće je na pnmer postaviti da prompt sadrži tekuć path što se postiže sa

PS1= $PWD' > '
axport PS1

Ovo je pnmar za *bash shell*

Pored definisanja *environment* promenljivih **profile** može da sadrzi i bilo koju komendu ili poziv programa. Tako u pnmeru 1 **setcolor** postavlja željena boje **clear** briša ekran, a **"eval = tset"** pokušava da prepozna tip terminala

Po izlasku sa sisteme **(logout),** *csh* izvršava datoteku **/ logout** Ostali poznatiji *shell*-ovi nemaju odgovarajuću proceduru

Mail

Ne všekorsničkom operativnom sistemu kakav je Unix, *mail* tj elektronska pošta ja jedna od najvažnijih funkcija Za slanje i prijem *mail*-a je zadužan program *mail* koji je sa manjim razlikama standardizovan na svim verzijeme Unix a i obuhvata osnovne komande za rad sa *mail*-om kao što su slanje pošte čitanje pošte..bnsanja redirekcija u fajl ili pnntar i sl Sve u svemu za korisnika veoma jednostavno Ali da vidimo kako cela pnče zaista funkcioniše

Sav *mail* za nekog korisnika se nalazi u direktorijumu **/usr/spool/mail** u datoteci čiji je naziv odgovarajuće korsničko ime Tako se za korisnika *paya* pošta nalazi u datoteci

/usr/spool/mail/paya

Svaka poruka je u čistom ascii formatu i sastoji se iz zaglavlja poruke (*header*) i teta poruke (*body*) U zaglavlju sa nalaze podaci o tome ko je i kada poslao poruku kome je upućena i sl Telo sadrži sam tekst poruke Ako je za korsnika stiglo više poruka one se redaju jedna za drugom po vremenu pristizanja U zavisnosti od verzije Unix-a poruke se razdvajaju uglavnom na dva načina Prvi je tako što se za početak poruka uzima linija koja počinje sa *From* S druge strane kod SCO Unix a recimo svaki *mail* počinje i završava se sa 4 Ctrl A znaka Na slici 2 je pnmer *mail* datoteke korsnika *paya* se SCO Unix-a

Sve poruke u **/usr/spool/mail** direktorijumu su nove nepročitane poruke Kada pročita poruku korsnik ja može obrisati ili je prebacih u **mbox** datoteku u svom osnovnom direktorijumu Struktura **mbox** datoteka je identična onoj u datoteci se novom poštom Poruka iz ove daloteke, kao i iz bilo koja druga se čitaju *mail* programom komandom

mail -f mbox

Za standardni *mail* program se na može baš reći da je prijatelјski prema korsniku Zato se na FTP serverime i BBS-ovima može pronaći mnoštvo *mail* programa (*mailer*) od kojih je jedan od najpopularnijih *Elm mailer* koji pored prijatnjeg korsničkog interfejsa nudi i mnoštvo zanimljivih opcija i neke dodatne programe, kao što je program za automatsko odgovaranje na mej Za Unix mašine na kojima je instaliran *X Windows* su na raspolaganju alrakturni *mail reader*-i koji konsle prednosti grafičkog okruženja

U poslednje vreme su sve popularniji i tzv *mail server*-i koji omogućuju interaktivno čitanje i slanje pošte sa udaljenog računara preko Inlemel mreže Kako to konkretno izgleda? Ako je na nekom računaru na Internet mreži instaliran *mail server* (jedan od najpopularnijih je POP *mail*) vama je potreban odgovarajući *mail reader* (POP *mail reader*). Reader i *server* komunicirau preko poruka (komandi) uz pomoć kojih *reader* šalje *username* i *password*-a *server* provarava ispravnost istih i vraća *reader* u podatke o trenutnom stalusu pošte za tog korisnika Korsnik će sada čitati svoj *mail* brisati čuvati u *mailbox*-u slab *mail* drugim korsnicima bas kao da radi na toj udaljenoj mašini Izmedu već očiglednih prednosti *mail server*-a treba istaći i da je ne ovaj način moguće čitati *mail*-e sa MS-DOS i drugih računara vezanih na Internet mežu Za MS-DOS postoje brojni *mail reader*-i sa modernim korsničkim interfejsima

Videli smo gde dolazi nova pošta za korsnike ali kako ona stiže tu i šta se dešava sa poslatom poštom? Ova pitanja zadiru više u oblast računarskih mreza nego u oblast operativnih sistema Ukratko poslatu poštu „preuzimaju" programi za rutiranje *mail*-a (MMDF *Sendmail*) koji na osnovu adrese na koju se *mail* šalje odreduju kuda će upuliti *mail* i kojim transportom Dalje pošta se smešla u odgovarajući *spool* direktorijum gde čeka na transport (**uucp, smtp, micnet,**) Poruke za lokalnu isporuku se smeštaju u pomenuti **/usr/spool/mail** direktorijum

Uređaji

Svim uređajima (*devices*) na računaru se pristupa preko odgovarajućih specijalnih fajlova koji su smešteni u **/dev** direktorijumu Svaki od tih fajlova zapravo predstavlja pokazivač na odgovarajuću rutinu u kernelu za kontrolu odgovarajućeg uređaja U stanjem sadržaja log direktorijuma primetićemo da na mestu dužine datoteke stoje dve numerične vrednosti To su tzv *major* i *minor* oznaka uređaja Naime svaki uređaj (disk serijska kartica video kartica) ima svoj jedinstven broj nazvan *major number Minor number* se konsti za identifikaciju uredaja isiog tipa Konkretno ako je *major number* serijske kartice 5 i ako ta kartica ima 4 porta, prvi port će imati *minor number* 0 drugi 1 itd Po pokušaju otvaranja nekog od uredaja kao i čitanja ili pisanja, pozova se odgovarajuća kernel procedura kojoj se prosleduje *minor number*

Pored *major* i *minor* oznaka, uređaj mora imati i svoj tip koji moze biti znakovni (*character*) ili blok (*block*) Znakovni uredaji su terminal serijska karta i dr. dok su blok uredaji *floppy* i hard diskovi

Specijalni fajlovi ne moraju biti vezani za konkretan uređaj već mogu biti i tzv FIFO (*named pipe*) spacijalni fajlovi koji se najčešće koriste za komunikaciju medu procesima Možemo ih posmatrati kao najobičnije liste Napravimo *named pipe* sa

mknod cav p

i pustimo jedan proces koji će da čita znakove iz *pipe*-a „cev"

cat < cav &

Zatim otvorimo za pisanje *pipe* "cev" sa

tee > cav

Ako upisujemo proizvoljan teksl u *pipe* "cev" isti tekst će bti pranet komandi **cat** koja će ga ispisati na ekranu Ovo je najjednostavnij pnmer međuprocesne komunikacije preko *named pipe*-ova

Svi spacijalni fajlovi sa mogu posmatrati kao bilo koji drugi fajlovi Sa njima rade sve vrste redirekcija upisa i čitanja kao sa običnim datotekama Tako na pnmer pored standardne komande **write** za slanje poruka drugim korisnicima, poruka se može poslati i na sledeći način Ako je vaš prijatelj na uređaju **tty07** možete mu poslati poruku sa

echo "Pozdrav!" > /dev/tty07

Poruku možete poslati i sami sebi upisujući poruku na uređaj preko kojeg ste ušli na sistem

Programe ELM kao i *POPmail server*-i *reader*-za MS-DOS možete naci na Sezamu u direktorijumu \U NIX

Za sva pitanja i pomoć oko Unix-a možete se obratiti autoru na adrese paya@fon.uni-bg.yu EPE KOVIC@yubgal51 bitnet ili na Sezamu na *username* ppakovic

Listing 1

```
SHELL=/bin/bash
HOME=/usr1/paya
PATH=/bin /usr/bin /usr/local/bin /etc $HOME/bin
PS1= >
eval `tset -s ansi ansi -s \(tset -s -s -Q`

Export SHELL HOME PATH PS1 TERM

setcolor 03_white blue
clear
```

Tipičan *profile* datoteke

Listing 2

```
qqqq
From branka Mon Jul 19 13:34 25 1993
From: branka@zver (UUCP (branka Radulovic)
X Mailer: 000 System V Mail (version 3 2)
To: paya
Subject: Jasi se
Date: Mon, 19 Jul 93 15 34 29 MET
Message-Id: <9307191334 aa07272@zver.UUCP>
status: 00

  Javi mi se popodnel

Branka
qqqq
qqqq
From root Mon Jul 19 13 46 48 1993
From: root@zver (UUCP (Superuser)
X Mailer: 000 System V Mail (version 3 2)
To: paya
Subject: hasi
Date: Mon, 19 Jul 93 13 46 44 MET
Message-Id: <9307191346 aa07282@zver.UUCP>
Status: 00

> Da li ti dogou ha mi promesis shell su bash?

  Done

qqqq
```

Primer *mail* datoteke korisnika *paya* sa SCO Unix-a

Posle letnje pauze u izdanju „Računara", avgusta smo se sa problemom prezentacija svega onoga što se tokom ova četiri meseca događalo na Sezamu – nije lako svesti gotovo 45 megabajta teksta pristiglog u Sezamove konferencija na tridesetak kilobajta ovoga prikaza! Od sada će „Sezam bilten" biti ažurniji nego ranije, tako da ćemo, recimo, u oktobarskim „Računarima" čitati šta se u avgustu dešavalo na Sezamu!

SEZAM

SISTEM ZA MODEMSKE KOMUNIKACIJE

23

Priprema: Dejan Ristanović a saradnji sa moderatorima konferencija

Moderatori: Dejan Ristanović (SEZAM), Nikola Božković (PC USER), Jovan Bulajić (FIDO), Dragan Ćirković (IGRE), Dejan Jakimov (PC PROG), Danko Jevtović (PC HARD), Slobodan Kalezić (ORKA), Renka Jovanović (RAČUNARI), Dalibor Lanik (ATARI ST), Bojan Pavkov (PC UTIL), Pavle Pekavić (PC OS), Dejan Ristanović (NOVOSTI i KOMUNIKACIJE), Darko Stančić (VICE/), Bojan Tepavčević (FORUM) i Zoran Životić (SEZAM)

U svom četvoronedeljnom periodu Sezam je „poslužio" preko 123.500 pozива komunikacija sa konačnicima 27.563 članova i preko 1153 dana (u proseku je 9.5 modema bilo **neprekidno** zauzeto) dužine je u chat-u provedeno 5255 sati i 658 različitih dana. Konačno su se Sezama dovršavale 4 sata i 7 megabajta podataka sata su za tolo vreme sistemu poslali 219 megabajta. Primetuje se dakle efekat 15 linija – broj poziva se povećava a vreme koje svaki korisnik provodi na vezi raznolike opsega (mada ostaje broj konačnika je je u zbiru broj stalno zauzetih modema bez prekidanja) je se znatno da se ne vaza osobi čim budu potrebna mnoga stvari brže efikasnije naročito jeftinije stave „kod kuće" na primer pomoću *Sezam del-ima ma-ster-a* (SOR). Broj downloadovanih u datoteka i spektro rasto – ako je Sezam bazično konferencijski sistem u uskoro ma linkovale maksimal ne se trudimo da dokumentujemo najnoviji PD/SW programa iz sveta tako da se vrednost prvih stvari koje su nekoliki sit zase ovakvom pristupa servisa nom se *Trickle-a* mogu prouzeti iz Sezamovih direktorijuma. Pri tome tesno saradujemo se nekim drugim sistem ma naročito se *Fido server-om*. Fakultetu organizovanih nauka (FON). Računskim centrom Elektrotehničkog fakulteta (ETF) *Imtel BBS-om*

Najviše poziva upute su Ratka Stojkovića (april) Irene Durmišević (maj) Čedomir Vičnjić (jun - jul) najviše vremena na vezi su proveli Irena Durmišević (april i maj) Ivana Miloradjević (jun) i Ana Ivanković (jul) najviše poruka su napisali Zvonko Marković (april) Goran Dimitrijević (maj) Vladimir Pekavić (jun) i Nenad Anđić (jul) dok su najveće konferencijske poruke poslali Aleksandar Šaković (april maj i juli) i Čedomir Vičnjić (jun)

Sve se konferencije tiče ustaliti ema se na nivou da preko 10 000 poruka i 10 megabajta mesečno – ukupne je pristiglo golove 40 000 poruka (43 megabajta teksta što je kada bi se arhivirale sitnim slevima tele knjiga formata telefonskog imenika) broje SFRJ u vile sistemu i 37 megabajta datoteka pristiglih ih uz nih 30 580 privatnih poruka (12 megabajta privatne pošte) 10 625 poruka u grupama (ostali megabajta lakote ne računajući datoteke priključene uz te poruke)

SEZAM — Zoran Životić

Za nevolju pliski tium poruka u konferenciji SEZAM ka vu je posve krivi su korisnici nazvali „mrtvinje mada 15" Naime nego prve kraj juna, jedan da računara ne Sezamu se pokvare. ko na jako tiudan način rađaje je sasvim normalnim sa je povremeno u neku od pisnici izbacivao znak dolar ($) S obzirom da je na taj računar upravo tada instalran eksperimentalni UUCP softver ove sitne greške propasval normu ostali sme menjal mrežnu kartu mašinu pisti Probiem je opstao dok kompletan računar nije zamenjen novim što je sve skupa pet kompletan računar nije zamenjen novim što je sve skupa pet kompletan računar nije zamenjen novim što je sve skupa pet

U tem *mesecu* diskutovali smo o problemima štampanja pod *Novell Like*-om pokušavali da *Novell* stanice ustanovimo ko drži neku fajl zaključan počeli o programiranju *NetBios*-a ksi o grcli emma se netalacijom *Novell*-a na mašinama se SCSI kontrolerima (Sezam uspešno radi pod *Novell*-om sa SCSI diskom od jednog sjajnija na jednom od servera) tek je problem saradnje TCP/IP i *Novell* mreže ostao otvoren

Bilo je reči i o operativnim sistemima (bilska) bukvalnost *OS/2 1.1 Windows NT* su paralelnjem u bilom svetu a kmisaci Sezama su zračavali svoja razmišljanja da im veliki na svetu mašinama Nedostatak kamkralnih iskustava nije spredao razgovor o svoj ,vel kog inci" čy sme ishod pokušali da pregnez rama Na *Usenet*, krajem jula su pojavila sumnja da ćemo uskoro prelaziti na neki operativni sistem posle je pozar u najvećoj lančni RAM čipova izazvao golove dvostruk skok cena RAM memorije koja je NT-u tako potrebna

Tokom ovog perioda konferencija PC OS debila je novog moderatora S obzirom da sa Aleksandar Damnjanović koji ju je uspešno vodio još od osnivanja, sešlo u Kanadu za moderatora je menovan *Pavle Pekovć (djakovac)* dosadašnji moderator konferencije PC PROG. Za novog moderatora konferencije PC PROG menovan je Dejan Jelovć (*djakovc*)

PC PROG — Dejan Jelović

Teme *algoritmi* je u aprilu obišla Kreta i ostala vetkane i poza tavole se prizemnim (ne svakako kersni mi) sivomna, kao što je poslejanje kandrenih cifara kod „zvanićnih" brojeva kakav su JMBG broj Jure računa brojevi na čekovima U meju je u ovaj tami bile male pouka izdvajamo kratku diskusiju o nalaženju en nmume numeric kih funkcija u više 4menzje a nemogucnosti genetsk h algoritama da se prizgove km problemom U juru smo diskutovali *n animacij* pokgona za ona koji misle da se tu radi o jednostavnom crtanju i brisanju značne pritanja i odgovore problem spada kal ko u domen naske tedke i u domen lokusiva i jhonva kake zavarati nke U julu sme se vratik „stalnim problemima kao što su algoritmi se serijalizje *quicksort* jeske brzinski šimjeri u nermalnim s tuacijama all čmjenice je da postoje specifični slučajevi kada se mogu premestiti i ćruju za te primene brži algoritmi, kakav je ne pr mer *radix sort*

Teme *mc dsej* je još traslicu bogata porukama U aprilu sme se „takmičiti" u pisanju Sta krsnog programa za *rasel* razmetni vieju da se pre programe SHARE slanjuje MAPK kaka ti a u slučaju „pada" neka se Windows api kacija dialeteka mogle aktjucak komandom RELEASE (pokazala se da ava rešenje nje savršene ej sme ga posllo se nye našeo neko ke to „pokahonase SHAPE EXE da maža da eskloci aktjucane dalateke na dalje jednvati) U maju sme prati t zanimlju diskusiju o *InDOS* liniji koj gover dit k ji ju kvirišava nje neke DOS funkcije Što je od veskog znacaja za one koj pišu interapt rutine TSR programe Takođe su se kaz h nalmi da se ekli vraju nestandardni VGA modovi npr ajnej sa 90 znakova u redu U juru smo diskutovali o mogucnosti „blokiranja RAM dielka „pleteli tevati grafcka karbas i odredili da t program radi pod uvedem ti ne U julu sme nastavik da „prepoznajemo i ig thaka (RAM hard fle ja) šta se pokazalo jednostavnim zadatkom za DOS funkciju 1Ch Fekvenzije i jedanje ubrzavanja *timer* tick nkarapta koj se zvršave 19.2 puta u sekundi al je ovuta utvrđano da je to sravl vo apsolv rane vec ran ja ta da sme treba pomazli po star i m pokukama U tematna koje se vezle programiden jezic me (jezic *assembler* basic pascal *cccc*) takođe je bile puno poruka – pričale se o BCD brojevima, grafici ca razn m verzajima GW BASIC-a vuđila se latka se večnim DOS ograničenjem sol najvrše 20 otvorenih fajlove u programu, C programa su konvertovan ne paskal videli se zatim fajlovi koje u *small* modelu generiše *Borland C* ne tipe pičmenu *Copyright* poruka a zanimljiva je b ke i objašnjenje svih *event*-a koje *Turbo Vision* podržava.

U maju sme se vratik veselej" rasprav o „najboljem" programskom jeziku ovoga puta je izneseni nez čvrstih argumenata za i protiv pojedinih jezika, ne varujemo da se neke zbog ahnja šte je tu pročitao jezici na neki drug jazik (stare navika se ijak teško menjaju) ali iznesem argumenti mogu dobro doc kada treba ocen t neki nev jez vac kay ce ekci jeste i nesmo objektno-orijentisanog programiranja tanesene su „jackal krkavi" za direktne adresiranje vrlen memenje a posebno oduševljenje je vazvale kansnik *sken* kay je u temu *pascal* prebacio „lene" kansnik UNIT-a namenjenih svemu šte vam može pa sti ne pamet ilen se da se stan san o programiranju kan sselavljenju raznih kompanenom jekaske ostvaruje

Tema *cacc* je dožvele gretku alegiejcju kada je jedan kolisnik ostavio letkl The case against C" k kemi se stvjatnavju sve mene C-a a kao primer „uramog" jezike se navod *Merlin-2*. Ma kol ko da velite C lokat pek vredi prečtati je je najvisan zaste mesegrane kakav je to ita u njemu pske ngramne je još ostaje da se utvrdi

U juru je bile desta psyolračnih prianja odgovore a mašin tayama *Turbo Pasksl-a* a diskutovalo se o palanju *Caps Lock* tampe što je malo problematično na stanjim tiNOS-ime U julu su „vruća tema" bile DPMI programi, nandote ane prsan u pamtenu Pesdal programam su se talvik formalnim stanjenje u memor e, odtavanjem dodatnih BGI drajvera i fontova i jesanjem TSR programa Tema *acc* nslv la se aplalvolcou prelaska se Packata na C++ i pisanjem interapt handlera.

Uvek atraktivna tekst *clipper*-i *baze podataka* su u čtavom periodu bile „najopleraćanija" nalše su se tu funkcija za rari sa YU Slovima tvrdnja da superbrzi *Blinker* pod izvesnim uslovima previ jesporne ekse nikramenkadu rinujeme kanstaci EMS i XMS memoriju (da l vam se fikrowse blakra? = ep sana je neophodnost kandelenja funkcija *txtWrite()* pre nje sa svakog znaka Qosti poruka pasvecene je i nekoj nezamenjivoj *HANFORUM Toolkit*-a o diskutovali smo o situacijama kada se ne može štampati peke mrače ty nejgre se mora štampate u lajt a ovda laj pesiati ne štampać

U maju sme diskutovali a integratsiu podataka i potrebi se reindeksiranjem a prijnje kompatibilnosti nevjerojatnija *Clipper* a 5.2 se sistem biblioteke mana ostaje je otvoreno. Oti gotov h bibloteka koje su pristigle na Sezam spomenema *scrolliar* za objekat *TBrowse* kae r b-bioteku za slanje telesa iz *Clipper* programa U juru sme nastavili sa ejes varjem problema kompatibilnosti *Clipper*-a 5.2 e posumnjali smo u njegovu performansu nemerile kada se radi se *FoxPro* mogla sma „ubrzati" sma dobil je tek sva *FoxPro* naredbi koje podleću *Rushmore* optimizaciji Bila su senumi ye iskustva s svešlama koji zbog tirine ne mogu da sasvim ne pay t h a nihovim štampanju u delovima i kasnjem „lepljenju. Poslednju reč je kao u mnogim dru gim temama imale influepa na kay šinnu daklar bata polja u progra mima? Ukratke uskore ne luete ojednomaje. On ne da ce kompljuten stati i da ee se ljudi vrati starcj papirnatoj D ev m problem me dis kutoval smo u juru kada nas je „mučio problem premene sistemskog vremena *Summe II Clipper*-a upravljanje objektna prekida na *Clipper*-a zaključi se su mnoge raspalože bibleteke napeu žitanje ver. 5m se nece tek kompatibilne se *Clipperom 5.0* (?) kay deda da uvede prave objekta

PC USER — Nikola Bošković

Tema *edition* i dalje dominira konferencijom PC USER! U aprilu sme upoznali razne jednosti *Word*-a konsnemca se posebno dopale što su ogranićenje svedena na najmenju mogucu me u na očitu kada se radi a dužim dalelcke koja može da se isp sn u Bilo je razmi prijanja o *QEdit*-u s spomenuti su i n odnosno emacs za *Unix* pokazalo se da je *pedit* od svih sedena ne može da se kanjunske da Jic na *QEdit* jmnogema ta ko dnevo deško) jer se kansk sacvem druga fikcijalja rada, ak se pod *Unix*-om može preci na *yee editor* čiy je sat kamanik *WardStar* kompatibilan U maju je kansnika *Brief*-a „mudile urevnavanja dsume margine koje jeske vel kog b ga pokazalje, razultira spadkom – cgljeda da je u jetanju lag koy se ređave pe vremen m upsom dalelcke s ne dek Ljubitelj *QEdit*-a nesu mogl „stu rate" da prokgementaciju svu menu *Brief*-a jer nphov editor ugađa i nemo način za uravnavanje dsesne noce Zatia je Darko Stanišić (*darsa*) napisao program IDM koj uravnava dsesnu nocu prezvoljnog teksl loja

U juru smo upoznali tasg programa LIST kay se ugnjava na diskovima U o jem nekl katalogu mume n jedne dalelcke U julu smo pokušali da nademe editor koj može da „uvuce lekal neograničene dužine *QEdit* „staje već peske 400-45€ K ja su nalenjam razni druge kandidati mesle kayma je i druge proicvedi zuitera *QEdit*-a forme Sem *Ware* editor je ne zalosi još u lazi bata testiranja Pamenuti su i *Brige* VDE sal ter kay se dnleya uz *Fortran 5.0* ertwerk. Program kay je preuzdan sa mreže da nešava svu problem je *WordPerfect-ov Program Editor* a pakata *Office*.

Tema *tekst procesori* dsenela je nekoliku zanimljivih dsenaka iz *PC Magazine*-a vezanih za neke problerne kay može konsnilac nevjeh tekst *procesora* a zatim je zapodenula i diskusiju o „najboljem" tekstprocesoru u *Windows* okruženju. Kada *Microsoft* bude ekine ova pe pedesnosti a internem radu *Windows*-a možda će neko druga firma uspeti da pun a ngtvorenn *WinWord*-u ali je za sada on ne malejnoj mnogih najboljo što se može ima naravno ogranićenj malejnjeg neki konsnici su na primer prezadovoljni programom *AmiPro*, za ka pe bnde da je brž i bolji od *Word*-a. Mnogema koj su pravachnovno zainteresovani za kreativno pisanje u poncipu smeta što su (?) *Windows* tekst procesori postrazumevanju tamna slova na svetloj podlozi, sto zne da smeta čak i oku je monitor kval tetan Delovanje koja nije problem al ima nek h boja koje se ne mogu menjati pa delazi do pojave kontrolne sinkam „nevidlj vi tasine okoline

Jul nam je donelo opsežnu diskusiju na teme Tex-a sazanak sme da ma nekih čvorava u svetskoj mreži kay se leave uskljuće evem programom mude vekin kroj konsnih dsetaske

Tema *wo rd perfect* dsenela je nekto manje burnuke nego ranijih meseci U juru sme proučavali sederšku stranim jezicima, kada se u manju dalu nše druge jezik nece se premenik poruke u memorne HELP vec samo formal esdanja. Separator tisuća Pared koja se može premeni poruka „continuest koja se spevje u lusnetama ali je ja mejuse samo u verzji *WordPerfect*-a koja je prilagođena našem govornom jadnuju Sve u svemu ako nemate program za kontrolu spelovanja prilagođen našem jezku (a sva je prilka da ga nemate jer je ne kompanija *WordPerfect* nje još ezsla), najbolje je da enlanete na amerićkom formatu.

U maju je napvše pazrje vazvale mlame kay okazalnaje konver nju tehničnog teksta u oničnu a u juru program kay dialazna pregled i klasifikaciju *WordPerfect* malenak U julu smo čtsh razne trafieve o (tudućem) *WordPerfect*-u 90 kg te trebae da radi pod DOS-om o da izgleda kao da je pod *Windows*-om da kanch *True Type* fontove Pasleđnh sme se i kreiranja sadržaj pomoću TeC manera. Tema *word*

	APRIL			MAJ			JUN			JUL		
	Poruka	Dužina	Dat.	Poruka	Dužina	Dat.	Poruka	Dužina	Dat.	Poruka	Dužina	Dat.
UPUTSTVO	0	0	0	0	0	0	1	1272	0	0	0	0
NOVOSTI	1055	3079843	508982	1097	4236570	32956	1028	3504890	159729	951	3885501	174540
SAJAK				99	50251	0						
RACUNARI	82	44203	0	452	399794	0	92	59821	0	50	31909	0
SEZAM	359	2338730	0	794	6071887	53172	1072	7707O7	50990	1095	7213988	81135
KOMUNIKACIJE	740	921482	897981	1033	797451	1058716	1234	872449	858298	1017	993929	989425
PC-HARD	389	300520	92435	387	304629	89209	407	295794	122934	571	424933	126651
PC.OS	405	308408	1026577	973	749870	2218794	372	256954	1781741	532	449320	753959
PC PROG	649	597558	1590075	899	794551	1087480	477	374150	954248	489	382898	588803
PC.USER	253	169137	377309	305	220951	392012	226	198895	349519	279	215807	957203
PC UTIL	285	219497	479117	378	276089	1970449	948	442892	705990	729	578255	933380
ATARI.ST	129	65551	363742	109	72828	550891	79	44021	21599	108	69403	579696
AMIGA	50	37508	127948	99	44520	7937	33	20791	34937	24	13643	0
ORKA	395	280525	189996	109	152924	379348	490	357739	0	179	119200	54314
IGRE	988	397149	784754	280	153581	239489	537	323008	190909	537	318991	10233
FORUM	542	929445	0	393	388298	0	769	1095489	110942	1118	1159443	38350
CIVILIZACIJA	579	524793	208311	1113	997379	557755	1353	1000329	580573	859	623051	172540
EXTRA	950	558901	38350	504	483188	759702	973	92854	92393	572	523411	477415
CHAT	168	103729	0	127	85994	0	97	60112	0	135	75520	0
VICEVI	299	205359	0	223	173773	12890	302	231970	57852	147	137050	998
MALI.OGLASI	598	330499	0	659	388321	19419	709	407434	19131	599	330918	0
IMTEL	99	95790	34026	93	39799	0	19	11258	0	12	8485	0
CET	21	12534	0	8	7997	0	29	22728	0	59	58598	0
NOVA.ZAJEDNICA							299	410271	0	23	29198	0
Ukupno:	8350	9599800	19689303	10091	11299522	9093583	11194	11959572	9090816	10043	10871192	5139436
Priv. poruke:	7390	2577435		7295	2829638		7398	2885224		8431	3683317	
Grupe:	2990	1012371		2091	884042		2717	1154892		2854	1239582	
Broj poziva		30102			32579			30960			29867	
On-line vreme		9900:08:12			7291:49:57			6887:22:35			9613:39:33	
Chat vreme		1239:58:42			1298:20:59			1354:19:21			1392:13:45	
F. treš. vreme		2797.24:41			3024:05:31			2793.09:13			2554:18:02	
Upload dat.		1174			1090			942			1199	
Upload kb		70443			59816			40949			90904	
Download dat.		30909			30949			30212			29533	
Download kb		1971344			1873275			1999127			1539977	

KAKO POSTATI ČLAN

Postopek učlanjenja poteka zvezno Sezama. potrebni parametre komunikacije na 2400 BN1 (ako imate MNP modem, uključite MNP) i pošljite #71)648-422. Poleg podatkovne poruke, Sezam še sporoči prijavo

Username

Na vamo je ime do otvoritve NEW (brez komarkov), a Sezam da vas daljo vodil brez prevne prijavljivanja.

Ime kraena

Jedan od prvih podatkov koje treba da date je ime ime i prezime, a potom i pseudonim pod kojim želite koristiti usluge Sezama. Jedinstveni pseudonim odlaze vaše lično ime na Sezamu (ne može se mijenjati). Ime može da ga treba pažljivo izabrati – predlažemo da se još jog prvog upotrebite ne pseudonim koji će biti vidljiv za posjetioca na vaše ime i prezime vidjeti nemate vidjeti. Sezam će vam predložiti pseudonim sastavljen od vaših prezimena i prvog slova imena. Pseudonim može da ima između 4 i 16 znakova (propustljivo se 5–8 slova, pošto vaše imenom znati vaše kolege, koje će vas tako i na ime koji se vama komunicirati) i u njemu ne mogu biti razlikovana slova i znak tačka.

Ime tajnog

Sledeći korak je izbor tajnog (password) kojim štite vaš račun – on sprečava od pseudonima koji je prima informacije, kojima znate samo vi i Sezam. Zato je izaberite pažljivi nemožite izbegavajte da tajnog kuda potreba pseudonimu ili da se poslije i i vašeg imena, nalazi ili imena rečeg dana pravilno – takve kombo se lako pogode. Ne donosi nekoliko pravilno pre svega znamo koji je tajnika „izvrsnim" sistemu koristio dosmet mešala da manjaja.

Ime raden parametara

Ostaje još da izaberete razne parametre (ažriver protokol i slično – ne brinite ako još ne znate šta je što, pošto će Sezam uvek predložiti uobičajene vrednosti).

Unos ličnih podataka

Prilikom unosa adrese i telefonskog broja budite vrlo pažljivi, jer nekoliko unesena adresa znamenjavanj vernosti Uprava Sezama se vama, a sasvim ion i vaše učlanjenje. Uneseni podaci da prevarite da li se podaci koje da uneli ispravni pazite sledeću dana, unesite isključeni pseudonim i adresu. Sezam će potvrdi da vam je jedne poslate ti zahtevi da dogurite podaci. Posle dobijte pismo koje da opisuje pravo na koji se pristupajete i na osnovu koga upisete upisano odgovarajuće stanje

do 1 mesec	8 bodova
od 3 meseca	15 bodova
od 6 meseci	25 bodova
za 12 meseci	40 bodova

Vrednost bodi je vezana za kurs nemačke marke i raste se pridruži dinamiku u bile časova jedinica. Na uplatnici koju dobijete da računa uplatni su, u kojoj upisi vašeg korisničkog imena, i trenutno važeći iznos za svaki pojedinačni prijatelja. Kao što da koji bode uvrsti uplatu po navedenim iznosima. Na primer: 100.000 -390000, 500000 i 800000 do 17.04.93. Ako uplatnjete ne svojim upisanim vrednost bode odmaknu ceni pristaje mašete stanak se Sezame ili ako se jedite upravi poslate

Nekon evršene uplate kolokvija potvrde pošalja pomoću na adresu „Računi" Vaše vrednosti Mikro 17. Beograd 6 telom na telefon (011)646-140 ili (011)647-966. Upravo će vam na dan prijema potvrde odobriti prvog posetnu, a sledećeg dana da na vašu adresu poslate Uputstvo za upotrebu Sezama, atraktivne štampane knjige (180 strana) pomoću koje ste upoznati sve mogućnosti sistema.

...i otpočnite učlanjenje

Ova kašjma se kun mogu poslati komarne Sezama na svoju *naručbu* ili. Javite se na ogaon nalin e ona, ne dokajati pomoć svetite uplatu u poštu ili banci navedeno podatke **Poštanski** vaše ma prezime i pseudonim koji da izaberi. **Primalac:** BIGZ. **Račun:** ili **Svrha uplate:** Pretplata na Sezam. **Žiro račun** 60806-603-23264 **poziv na broj:**06 100-5. Potvrdu o uplati pošaljite faxom da 13 časova (računi danom) već u popodnevnom časovima da pomoćnik. kominik Sezama!

Dvosmisleno prijavljivanje

Ukoliko lokom predstavljanja pseudonima (npr. vesta se prezime), pozovite pomoć i pojavite se, navodeći kompletne podatke **Podnijo:** bilo da se prijavili dan ili vaše pute, kontakte ena kompletno ime koje je napisano na uplatnici.

Kodiko vaši uplatnice

Cene navedene na uplatnici vaše same da navedenog roka. Novim pretplatnicima daljeme za uplatu rok da pet dana od trenutka prve prijave a stanu pretplaćenima od prvog upozorenja na produženje pretplate. Smatrame da je pet dana stanim dovoljno da naše uplatnice stignu do vas Uključili ovoliki uplatu u ton roku, pretpostavljamo da ste vaš pristup na Sezamu koje vaše na dan uplatu. Ta ne znati da se mjenite pravovo prijavljivali dovoljno je da potvrdite Sezam, predstavnik se obaviznim imenom i lozinkom i sistem će odmah ažurne sano pretplate.

Ako upisujete na svojim uplatnicama

Za uplatu koristite originalne uplatnice koje dobijete da razdajete. Ako koristi da uplatu vaš sveštite na svoju uplatnicu, cena koja vidite prilikom prijavljivanja na sistem odmaknu prilikom prvog upozorenja da vam pretplate ovoliko važe pet dana i da date koja vaše cene na dan uplate.

Raspoloženje sistema

Ukoliko pre prijavljivanja želite da raspoložete sistem, na prazne **Usernome** odgovorite na DEMO. Nato ada se u demonstracionom režimu rada u kome možete da pregledate kompletnu strukturu sistema koje je kreirana „pravej" Broj DEMO pomoć nije ograničen, a vreme se jedno trenutu je 5 minuta.

Terna *arhiven* denasa je znatno manje poruka nego pretkodnih meseci (izgleda da se korisnici u prethodna tri meseca izmili sve (8 bez vezinu) argumenata i umrzli brzine rada programa za arhiviranje tako da se težnja diskusiju premele se programe koji omogućavaju podelu velo koji datoteka tipeta je jedna zanimljiva ideja o kreiranju velodelnih arhiva PKZIP-om bez poseduvanja diskete, na primer na RAM disku (i preuzimanjem određenih DOS pravna sve to je celina u domenu tekmičko razgrava Konenca ARJ-a u toku su otočeki nova verzija (2.41) koja te trebale da konkurdne verzij 2.04 PKZIP-a. Tim povodom je aktuelizovan problem kopasposobnosti različite arhive – zaključili smo da bez obzira da arhiva kreirana PKZIP-om ima na kraju centralni d rekterum koji te moguć da povezi izgleda se oporavak PKZIPFIX na ume da koriste ova podatke pri oporavu" lokalnih zaglavlja, pa se sve svodi na mehanizme koje i ARJ ima.

Zanimlja je bila i raspirava o pseudonasih šifri koje su ugrađene u popularne arhivere (npr PKZIP -s) – pokazuje se da se kendšeno mekoli sasvim dovoljni za amatere ali predstavljaju lako rešiv problem za šiokivanog kriptoanalitičara. Konstatovano je da za relativno popularne arhivere delimi konzibe specijalne programe pomoć kojih se to (npr PGP) kao i to da prezbacivost izmenoj koda znatno povećava povezenje u sam program. Prehoje je nekoliko slučaj koje se desile „iz arhiver – *ZZap 6/6* koj komvrtuje jednu arhivu u drugu *SHEZ II* i *WinZip 4.1* mnogo drugi

U temi virus kementisanje su nova verzje SCAN-a. CLEAN-a (poslel sme do verzje 101 ! stoji de 106 s krajem jula se pojavi i *NetShit 7.03*) prverena kuna opezorenja koja nas da bh programa dajo a nemogućnosti da se zaštitimo od virusa. Virus *Tormentor* koji *shell* za rad s popularnim antivirus paketima. Opisan je novi virus koj se pojave u Slajeniju koji tekuci SCAN na prepoznaje otkljenjen *Yankee Doodle* konfiguranjem programa debug upozorenja na virus koj se koje u ongnenisu arhiv kojum je temi sveta distribucan *Telemate 4.11* e nacmejal sme se veoti da su Albanji saumali (!) prvi domaći virus.

Mogućnost zaštićivanja animacije napravljene u *Animator Pro-u* direktne iz DOS-a bila je važna komponenta terne grafika. Raspravlja le se o prednostima i manama programa za kompresiju skoka iz jednog trenutka u drugi kao i gubic.ma o kvalitetu ukoliko se koriste JPEG kompresna. Konstatovano je da je jedan od najboljih programa za kompresiju *Alchemy* čiji je autor Hanni Kjaa se može skoro gigantskim dimenzija slike koja se može konvertovati. U nekolim temama diskutovali smo o mogućnosti kreiranje „telefonog programa kao door-a nekog BBS-a (mogu i se uiaz i izlaz pouzdane preuzmenti se modem?) mnjoja sme direktorijuma prelaznoda

Iskot iz *Words-a* i *Carol Draw* razmatrali posledice tne levo formatora na IDE diskova. Posledice smo otkrili pomoći programa BV koj omogućava pregled vrhednih API API drivo i pregled arhiva u arhivama (autor je Bojan Petrinić (ebrjen))

ORKA Slobodan Kalezić

Posle uzivajanje konferencije IGRE koje je nekada bila „samo" treba u ORKA konferenciji tanj peruka u ovoj Opšta RAzmenska Konferenciji i dalje se drži se tradicnom novu. Naročito vredni pažnji stizali su u temu *electron*. U spisu je Dame Tadić (*dtadic*) poslao poslao adresa i lokalona „uvih radno u Beograd koj imaj kakav-tekuv ubre kompovenata" ste je razvrale mogu Vu Nemačkoj. U mayu je Robert Perović (*robert*) poslao projekat ie samogradnju visegentnog regenerator i brzog punjača NiCd akumulatora iz majskog *Chip-a*. Projekat je hardversko-softverski pumjerni uzivaja PC računar vedeo radiona o stanju baterija. Nadena je i aelekvatna zamena za neke delove koji nad nas nema u slobodnoj prodaji. Ukoliko je kandidatu programa za praktikovanje štampanih pločica. Stigao je još jedan simulator mikrokontrolera 8051

U junu je bilo dosta odgovore na pitanje gde „nestaje" polovine energije naelektrisanog kondenzatora kada mu se paralelno veže još jedan ali lakav. Zanimljive diskusije u ovoj tem bile je i u julu, kada sme se ozbiljno bavili skretnicom na telefon ij, uredajem koj kao dva paralelna vezana telefona isključuje jedan kada se drugi u upotrebi. Stigle su dve šeme jedna sa relejima a drugu sa tranzistorn drugo rešenje (poslao je je Zlatko Sučić (*gerry*)) ocenjene je pogodnijim zbog svoje mimjaturnosti (staje u utičač za telefon) i odlicne se nanavkom relaja. Dobil smo pri ovo šeme uredjaja za uključivanje računara preko telefona i saznali šta se može vrsti iz ranake tranzistora.

Teme *otvaranje* u aprilu donesla prve vesti o nastavku razvoja kalkulatora HP48SX – to je HP48GX (poslaj i sramotnom tehnu HP48G) pri čemu je cena još kao što je bila bila za HP48SX. Nekoliko dana po premiere pojavili su se tehnički detalji o ovom računaru a bile je i „neposredni izveštaja" se prvitelnom koj se mogao videti u Nemačkoj. Glavna problema je izvestuka brži probesor sa 32-bitnu sabirnicom i nov kalkulator za koj se kaže da će imati veliku memoriju. U letoj tem nagasi sme i konverzeme HP41 i *Sharp PC* koji su ostali nas upustivu, u „srednej" tem *hp48sx* vidi smo kako se uz pomoć privlačnog i jednostavnog programa može relativno brzo razviti neke nestandardne kalkulacije za mvaj kalkulator

Teme *otvaranje* je lepo startovalo, u aprilu sme setali zadatke sa republičkog takmičenja mladih programera kao i listu najboljih. Je sedi Kratica (*kratica*) nas je predstavio svoju verziju programa memikoj Olimpijadi koja će se održati oktobra meseca u Argentini saznali sme kako se brzo zadeci se Olimpijadu kada se formira komisija za pregled kako se bodovi zadaci kakvi je program bonavka i takmičenja. Ekipa je određena se majskom Saveznom takmičenju u kojem su „Računari" već pisali pešte su sva troyca pobednika konkursa Sezama Uprava sistema dodala im je pokrajinskoje prelaske u uku je trajenje sponzora za put u Argentinu. U junu smo dobrotvak o njihovim pripremama, kao i zadacima sa desetadnjih Olimpijada i se srujili takmičenja. podsima naših ekipa sredan put mnoge uspeha.

Teme *otvaranje* nije se naravno, bevila stanu takmičenjem programera – diskutovali smo i o tesk sta te v računarima kantici nauči u srednji škoh i se kojim tvorčem časova, da li je za dobrog programera potrebno da dnete poznaja matematiku (mišljenja se bile vrle različite, ali toga da se sve može naći u knjigama onda kada je zaista potrebno. da toga dil je matematika vema potrebna i da programer može potpuno da zemnadi prihvaljive rešenje bez je ne poznaje) Vdiek sme i kompletan program informatičkog smera Fakulteta

organizacionih nauka kao i programe ranih američkih škola. Za raz liku od naših programa, učenik ima može da izabere oko petnaest predmeta što bi baz ske se cena je diskusija o temi treba i teraju učenike programerskih smerova da čitaju ruske klasike (tzv. „ave ruske priče") što veoma dobro prihvaćena u našim školama.

U tom zakonu ima u julu zatim zanimljivu diskusiju o GIX-u koja je nastala povodom u Interjekte programa za informacije poslednja PGP od strane upravo lej pozvanog američkog sistema. Ovaj program je bio dat u sorsu diskutanti su smatrali da je ovakvim ponašanjem BIX-a povređena njihova slobode govora se drugo strane američka vlada je menjala firmu samo da ih potencijalnim zaštita ove algoritam otežala njegovim širenjem. Uspešan emulator starog dobrog *Spectrum*-a za PC računare ozbiljno je na svetlost dana radi e zabavljanja igre i naših zadnje *Spectrum*-ovih programa, kao i zakonske strane oj hevog konkurenta. U julu ama u ovoj temi poklonih da saznama šta trenutno važeći zakoni kažu o carinjenju računarske opreme ugleda da prilazi podnošenje odstupa od propisa.

U temi *literatura* traženo su i (ne)viđenom) nađena razna uputstva i knjige. U tome najbolji bio je veoma male poruka jedina zanimljiva diskusija vodila se o *Fidzar*-u novom Atarijevom računaru. U temi *help* upravnja je mnogu net pridružen ja PC disk jelenica na *Commodore 64* (pouzdanog zaključka još nema) još jedinom opisom dejstvo Mikelanđelo virusa i ornasen jedan važnu savet za vlasn ke originalnih IBM PS/2 računara **obezvezno** sačuvajte sve dokaze koje ste uz ngh dobili (računare, kablove, kasetofone, ploče, bez njih **ne**će moći da izvrše i setup i pravilne eventualn novi disk!

Sredinom jula u konferenciju ORKA „preselila" se tema yu.kev (ranije u konferenciji SEZAM) u kojoj se diskutuje o problemu YU sleva, njihovoj ugrađnji na razn m sistem ma standardnim raspored me sleva, problemima sa popadnim programima, itd. Kraj julu donese je zanimljivu diskusiju o tem razne jedan korisnik je tražio neke savete za dramu koju piše u kojoj će se pojaviti „pametni" u neki kompjuterski sistem. To je navelo nekoliko korisnika da iznesu razne ideje i iskustva pe tom prilaju.

IGRE — Dragan Čitaković

Prelaskom meseca su u konferenciji IGRE stvorili preko 2000 poruka, što pokazuje da je stvaranje ove konferencije predstavljalo pun pogodak. Tokom aprila u tane avantura najviše poruka je bilo vezano za igrama nove generacije igre nazvane FAUS. Autori igre Srđan Veljković (*bleskiyanoša*) i Darko Milutinovac (*demo*) napravili su je kao odgovor na već čuveni SUP! U julu je stigao „ubica na ulici" u vidu SUPI-ja 4.0 Stanislava Sekerića (*sekiranac*) ovega puta već zaćutali bio je da inputujete u FAUS grad i da tame pravite razne pakosti kako biste smanjili prag igre FAUS.

Bilo je reći i o stranim avanturama naročito o igri KGB koja pokazuje da zbilja nije lako biti špijun. Zena zabrane igre nije vaš la imad tame *simulacije*, pa je talentovane sazamevaca u borbenom le tećmu uspešno prevazišli kraju. U ravnoj borbi Marko Knežević (*Jarcz*) je savladao Stanislava Sekerca poraženi su trabit revanš, a da bi sve bilo regulamo izvršenje novih pravila je obavljeno javne u Klubu programera. Igre bitu i nastavice beranca za program *Formula 1 Grand Prix*, jer se desila neočekivana i podsticajnoj bolida i najbolja taktici. Odmah se narazne povela raspravaa i tame da li se igre staze mogu bi ate vozih na *Amigi* ili PC-u kao tome da li je igrac upotrebljava algoritk ili na bile je reći i e varnasti simulatore u zavisnosti od brzine samog računara. Ipak i o upotrebi zvučnih kartica radi dočaravanja „prave" atmosfere.

Tame *strategije* odlično je sve tekstove vezane za drugu nastavak popularne igre DINA dva je u tem *legalno* dogodaj aprila mesa bio pojava PC verz je poznate igre OXYD arcade/strategije je desto spa ke kada se pokazalo da je za pretrazak ove desetog nivoa neophodna kupovina knjige. U julu je igra *Incredible Machine* bila u centru pažnje. Nazime sv ome gledali kako u crtanim filmovima Kosta Kijeli pravi nekakve mašine kako bi da jednom Pikcu Tračcu (vrsta mrave slično „Lemingsi da napravi mašinu koja će ostvariti tražen zadatak. Naravno raspoloženi Sezamovci se odmah reši dela se „prestekarsa" nove kep pravo najviše problema.

U tem posvećenoj mogućnostima igrama premnavanje je nova verzija demoboy Tetrisa ZBTET Auten Dragan Zakić (*dzakic*) Dejan Rusić (*deksi*) su u verzi 3.0 zorel mnogopomnih opcija za poboljšanja ne dostaj mogućnost modifikovanja tabmenaja dva igrnica Sezamov ci su se sličnje takmiliti u ovom novom Tetrisu. a bio je i onih koj su pobeđivali uz pomoć tastatura „transrah" posebnim programima.

Ljubitelji FRP (*Fantasy Role Play*) igara su u svojoj temi burno raspravljal na same o kompjuterskim FRP igrama već i mnoga koje se igraju „uživo" naročito na američkoj FRP scen. Pojavila se ideja o osnivanju domaćeg FRP klubo za koji bi treba naći prostoriju. Na veći broj poruka je ipak povodom drugom nastavku poznate igre *Prince of Persia Shadow and the Flame*. Najpre je stigla demo-verzi je koja je „unhurdak dahieve" a onda su mnogi mal prilike da spra bajuu i „pravu stvar". Poše ova igra ima elemenate kako arkade tako i avanture poruka o kojima su se iznosili i rešavali problemi su prob zole u mom teme ali ih ulje predstavljen jeki naročiti problem – u veoma kratkom roku uspešne su savladane sva prepreke i u interesantnoj borbi uspeli da vez *r Jaffar* osvojena nakloncol lepe princeza. Oficir von je i Jaaji lična pratnje. Koji omogućava nesejjivim igračima da bilo mnogo mukte pogledaju i odigraju sva neva. Povoljne reakcije izazvala je pojava igre *Flashback* koje je sličnog tipa kao i *Prince* a pe mešanja mnogih ga i prevazišao.

FORUM — Bojan Tapavčević

U konferenciji FORUM bilo je manje poruka nego cani h meseci posebne u aprilu i maju bahe se znalo dogodi da za čitav dan na pristigne ni jedna poruka u ovu konferenciju posvećenu poštu. U apri lu su najstrašnije bile svakodnevna propadnika i simpatizera razni opozicionih političkih struja o telo je desila sementara vesti stranih agencija iz konferencije NOVOSTI. Kako je vreme odmicale te ras prave su se povukle u drugi plan pred mnogobrojnu skarne vojne interesuje u Bosni dok je kraj meseca proteklo u raspevanju i pregle žama pe pisma Vase-Ovomovog plana.

FORUM je nagio „živnel" krajem maja i početkom juna pre sva-

SPECIJALNE USLUGE

PODSISTEM TANJUG

Podsistem Tanjug" omogućava pristup profesionalnim servisima svetske agencije TANJUG Rarli se o aktuelnim političkim ekonomskim, društvenim naučnim, sportskim i drugim vestima iz zemlje i sveta. Ser vis namenjen prvidu na Sezam prate stalne unapređene linije o najvažnijim temama iz prema potrebe mog. Ulaz ili pranosti na svoje računare rad dalje stranice. Pristup svim podsistemu omogućuje se iz menija (MENU TANJUG) ili komandnog meda (naredba NEWS)

Pristup podsistemu „Tanjug je posebna usluga Sezama i plaća se posebno. Za detalje ako pristupe ustava tarifama obratite se Novinskoj agenciji Tanjug, Kontakt: Milena Sekulić, tel: 625-722.

PODSISTEM BETA

Podsistem „Beta" obezbeđuje praćenje domaćih događaja u elektronskoj formi. Časopisi preuzeti na Sezam pre ili u trenutku otvaranja na kiosku dostaje se prema potrebama korisnika mogu pratiti dnevni, dnevnih na najvažnijim rubrikama. Pristup svim podsistemu ostvaruje se iz menija (MENU BETA) ili komandnog meda (naredba BETA). Trenutno se na ovaj način distribuiraju nedeljnik „Vreme" a pristup je u prisutnom periodu demonstrativan.

Pristup podsistemu „BetaPress" je opcijona usluga Sezama i plaća se posebno. Za uslugu ste prijavite i sakom kontaktiranje obratite se Mar kalingu Beta „Vreme" Kontakt: Vojislav Mićanović tel: 646-079

MOLIMO, SAČEKAJTE

Poslednje izdanje uputstva za konćenje Sezama je iscrpljeno i od 28 jula se ne isporučuje novim korisnicima. Očekujemo da će novo izdanje uskoro iz Stampe krajem septembra nakon čega ćemo ga poslati svima koji ga nisu dobili. Elektronska verzija trenutno aktuelnog uputstva nalazi se u direktorijumu NOVOSTI pod imenom SEZU PUT TXT i može se preuzeti komandom **download \novosti\sezuput.txt**

ga na dan smene Dobrice Ćosića s zatim povodom demonstracija i hapšenja Vuka Drašković. Uz mnogo uzbuđenja ogorčenosti (prem da ipak manje nego 9 marta 1991) korisnici su raspravljali o demonstracijama ubistvu maloletnog postupanja policije prema građa nima kritike predstavnika vlasti njihovim haranju u zatvoru i u bolnici neopsednosti spazenje da na tive iz inade odgovarajuć odgovor Ove diskusije trajala je tekom čitavog juna politickim pita, posebne predstavnika činjenicom da je Drašković 1 jula stupio u štrajk glađu do smrti i ukolanja prtvera. Tenzija je rasla iz dana u dan odlučio i ubistvoji Sezamovska populacija je dočekala se velikim nakljaženim i naradnih nekoliko dana se naširoko raspravljalo o mogućnima i ulnicima tog čina, o tome ko je tu zapravo kolko dobro e ke kakve izgube sil. Vremenom je rasprava na tu temu polako gubila na interesantetu sve dra ree potpuno potisnula raspravama a dramatič nom sledu marke i cena što se poklopilo se otkupom svega određenog roda šta Deselebrutne uvecane vrednost marke u reku se 10 dana podešava je jedna kolekcije predviđaju za kolike vremena će jedna marka vredeti 10 ultimativ dinara. Optimističke procene su predviđale godinu dana e pesimističke mesec-dva

CIVILIZACIJA — Dejan Balinda

Konferenciju CIVILIZACIJA se sve više sformira kao najpozna nija na Sezamu tokom sve šeste meseca u nju je stigio gotovo 4000 poruka, tako da je otvaran njen treći tom CIVILIZACIJA 3 Mo žda je za to zaslužna: uzavrela političke scena sve zamreni svakodnevnijeva diskutanti su se više oklame temama koje su bliže umetnosti. Ovaj novi me manat je obogatio svežim razgovore u kojima su se tada de minirali potencijalni konvoi u utv uzbijanji terena Nagovešta nja teme je *muzika*, premda je često merala da vodi „borbu" se jemat se temom *kim stav* sil je na kraju interesovanje za muzičke razgovore ubedljivo nad matelo na momente čak i jednu diskusiju o nasonalizmu i ne govim demetima.

U aprilu sme u svoj tem predstali jednu dirljivu priče o grupi *La Strada* u junu su sa kupljeni telefoni i stvarali zadih „esamnteh demajlih i stranih pesama a u juli se pojavile ti rake interesovanje se sve vrate muzike od Baha Mocarta, pre ka Boba Dilena, pe do najnovijih hitova – sve to uz stalni pensel mvea dela. je

Temu *uzrelost* u aprilu je donela koristen sjusak literatura ze studentie cetinja umetnosti s sep o poseti muzeju Salvadora Dalja u Figarasu nikoga nije ostavio ravnodušnim U teme

književnost je posle izvesnog zatišja pružajo nekoliko interesantnih kratkih priča Tome filozofije je u julu poklonila da rezmetin očnog letočnjačke i „rade (hrst'-anske) filozofije Manje je poznate recrne da se tik Platona nalazi na jednoj freski u crkvi Bogorodice Ljevške u Prizrenu izvesne uznenađenje predstavlja pričlna upomeni pri raz matranju nekih beogradskih pozorišta Fedrina Nikoja da li je zaista tamni lam lunjat i kakva se bila neka njegova druge interesovanja mimo lica inje?

Uvek zanimljive jezičke rasprave u temama o *jezicu* i rečnik u junu su se svele na raspevnu o stamelovenskom jeziku e mnogo nas je i poreklo pojedinih reči U julu smo pročitali zanimljiv nevri na veliko jezik, o temu rečnik nas je ohrevaj mesecu zabavljala nevrin smiksima se rečnik žargona. Ako se na sve to dodaju repertoari beogradsk h pozorišta poruke o filmu (posebno je interesovanje o filmu „Evropa") sećanja na putovanja u Bugarskoj i Prag i maštanja o putovanju u Japan posebno na naše veličanstvene Nikolu Tesiju. Mesta Crnjanskog razmišljanje nad pesmom Vuka Drašković analiza filmografije Klinja Saleska i Orsona Velsa, kao i polemičke rasprave u temi *hrm stav* vezane za ukidanje smrtne kazne izvođenje vannastav ke u školi ako se dobar ovaj u dno ste se dešavalo u ovoj konferenciji Ogromna bila poruka i velika raznovrsnost teme ali pravenje ekstremne poštene zamešnje zadatkom U teme ni vam mesta pomoći praksa jedne grupe aktivnih korisnika koji se na sebe uzeli zadatak da jednom mesečno napravi ekte najzapaženijih poruka – čak i oke neke emak vremena da pročitaje ove prekrasne ure ste je bile najzanimljivije!

VICEVI — Darko Stančić

Ni vreva u telefonskoj konferenciji nije nedostajalo Najvide (je veoprilske) diskusije (koja je svetu u konferenciji VICEVI zabavljena, ali se konačno snašlo pa uz najveću pokušaj o po jedan vic) oživljaje je pitanje „Po čemu se poznaju hulgeni u Matematičkoj gimnaziji" i odgovor „Po kome oni pozdravljaju ujutro u školskoj svečko" Dio je smeha i oka „naučnika" zanimljivih konverzacija, na primer kamikaza (auto autobusa) sanjive (sekjanke) superman (radnik Jugoslavije) sreo (san vanzemaljca) osigurač (sen mutl) bekalak (starački obdaništa) čačkalica (kazmeračka)

U maju amo dosta dva fikasa linderaševog nadu posvetila Laž nj predsednik Šimpan Mik s Nazivu O radi oglasa autora Jugoslava Ćosića i Tajane Nikolevskih Sile je zahvaljuje spasnuti se vremenu računarskoj mekušinu iz ugla jednog oka vlucetne obrazovanog, srednješkolskog profesora. Razlika između 286 i 386 je u obliji transistora u čipu 486 me u sebi devojčice transistora da se popuni pole ukoncat ili vina, a prvde me vrte vela desk. Čue sam da se sprema 586 Taj čip da emah veoma mečan disk svrake kas kalendara I kaise se lu sirija egramen bez linida, dioda i transistora i dobre se maleoko integrarne kako Deda sad tele ostav sterijski značaj"

U junu je na vrhu konferencije VICEVI došlo do kadrovske promena – Željko Krstić (*zkrstic*) dotadašnji moderator preuzuje se Vladimiru Krstonošiću (*vkrstonosic*) na funkciju ttv moderatora u razgovoru na Kipru a zamenio ga je Darko Stančić (*darcan*) Dobili smo i sjajnog novajcu, lekara veerva o plavušame (tačne 412 vrsova!) U julu se najviše pričalo o Crnogorcima i Lalema i Sosama, ali ni drugi nisu bili zapostavljeni.

Za kraj evo i nekoliko vrsova koji se pobedili na mesečnom takmičenju „Dešao grobar kuću sav zrnjav pe ga žena pita što mu je. On odgovara. Ma pusti tipu umrla kaše ti pula. ama je vasilli na sel" II iz mapa Seki Muje na kvasi sunce sve, plave cvrkucu i uhvati on dla se semezadovoljava. Kada je završio zapali cigaretu Vidi ćim puzan plaćena kod ugleda komerca kako gnjavio Marina Mujo rukom i ter kernarca, olenke ruka e pantalone zamišljeno rekle "Što ti je život. Seks, droge, rasite" Premenmo i jedan iz teme *bez veze* koje je još kruna nasa vreva u sklu jeku ukinom Sarajka dve frajere pe je jedan muve u drugi je pčela" ovaj lekar da te razumeli Ulazu bile kojeg časopisa sem Našeg „Na ulici dve babe pesle dve a druge SCSI"

Knjiga u elektronskom obliku au jedna od najkorianijih stvari koja se mogu naci na Sezamu. Ovaj put cemo vam predstaviti nakoliko takvih elektronskih izdanja, koja sva viša postaju nezamenljiva, a obzirom na ava manju dostupnost kvalitetna (iii bilo kakva) računarske litarature na našem "tržištu".

Zbog letnje šeme izlaženja Računare in meseca su protekla bez našeg redovnog izveštaja o tome šta ima novo medn delatelcame na Sezamn I uopšte u svetn sharewere i sličnih programa. A novitete na Sezamn je bilo zaista dosta za In meseca u našim direktorijumime sa našio 260 novih programe nknpna dnžine arhiva od sivere 30 Mb

Najviše novitela bilo je n maju i moramo sa pohvalI I de je to dosadašnji rakord u prilivu novih programa na Sezam – stiglo je nkupno 111 datoteka dužine 13 6 Mb Tokom prvih letnjih meseci je taj priliv bio malo smenjen ali ja dalja ostao zadovoljavajuć n jnnu smo dobili 86 arhiva uknpne dužine 7 5 Mb e u julu takode 7 5 Mb n 63 arh ve

U ovelikom broju novih datoteka za svakoga se si gurno našlo mnogo loge zanimljivog Bilo je tu raznih uslužnih programe elata za kommnikacqn bibioteka ze programere Windows programa Igara ali je najviše bilo razn h informacije

Ubedljivo najviše novitela n protekla tri meseca bilo je n direktorijumu e \ibmpc\info\ odnosno skraćeno IN FO U lom kalalogn mogu se naci razne informacije n obliku elektronskih knjige relatanih vodiča leksluelnih baze podatake recnika I slično Arhive u ovom direktori jumu n posledlje vreme sve više dobijaju ne znacaju jer je ovde postalo praktično nemoguće nebavih bilo ka kvu literaturu ili naku drugi izvor informacije vezan za korišcenje I programiranje računara

Sudeci po top listama najtraženij h programe u pe riodn od maje do jula I delje su najpotrebniji bl programi za zaštitu od virusa I njihovo otklanjanje (*Scan Clean VShield*). Takode je veliku pažnju izazvala pojava novih verzije ARJ arhivere DSZ/GSZ protokole za prenos da toteka modemom I komunkacionog programe *Telemate* Narevno vrlo tražene su bile i igre među njima i nekoli ko domaćih

Na našoj top listi najtraženijih datoteke objavljujamo samo podelke ze mesec jul De b smo I pored letnjeg opuštanja ovoj listi dali koliko-toliko ozbiljan karakter sa nje smo izbacili igre programe za zabavu

Kade spominjemo dogadaje vezana za datoteke na Sezamn n prethodnom periodu ne možemo zaobici ni veliku diskusiju koja se vodila u konferencq SEZAM Bil smo zakljucili da je višedelne ARJ arhive bolje obeleža vati ekstenzijame koje poc nju od A01 (znaci A01 A02 A03) e ne od ARJ (gde ide ARJ A01 A02) Ovo je pre svega logične – mislo je čudno de treci deo neke arhive nosi ekstenziju A02 a de A04 označeve pet deo a i de lepše izgleda je isps n ja naglavacke (npr prvo drugi i treci deo pa tek na kraju prvi) Al našlo se I onih koji nisu delli naše mišljenje a pošto su imali najgrlek slobodnog vremene rezultat je bio više od 500 poruka oko ovoko banalnog problema i potpuno uludo potroše-ne vreme

Kako nas pak nisu ubedili) da treba b ti konzerva tivan e ne logičan od sada će višedelne arhive na Seza mn biti obi ke file adn gde je n redno broj dela Na našoj top-listi ovako arhivirana programe prepoznaćete po .a02 ekstenziji I ovo rešenja je samo privremeno dok ne dode do novih verzije Sezam software kade će veliki program stajati u jednom komadu a svako će moći da ih podali prema potrebi u svom PAD n

Knjige

Knjige u klasičnom papirnom obliku imaju dosta prednosti u odnosu na elektronska izdanja glavna je što se mogu čitati i u ležećem položaju) a i za oči je manja naporno gledanje n hartijn nego u monitori Sa druge strane dobro organizovane elektronske knjige (nrp one koje su kreirane po *HyperText* standardu) su nesporedivo praktičnije – mukotrpno listanje Ireženje naslova i pretraživanje po indeksu u pravim knjigama zamenjeno je fleksibilnim pretraživanjem teksta i trenutnim unakrsnim referenciranjem

Ali sva i da još uvek vilše volita papirna od elektronskih izdanja Iu je jedan problem na koji namata nikakvog nlscaja – strane knjige iz oblasti računare se kod nas praktično više ne mogu ni naći a domaćih knjiga li pravode je sve manje Čak i kad se nada nešto vredno kupovina cena knjige je ze mnoge često poslednja i ne premostive prepreka

Zato dobyajn ne značaju elektronske knjige I zbirke informacija koji sva više ima na Sezamn Nake su u obliku obćn h ASCII tekst datoteka neke u obliku doku-menata spremnih za štampu iz nekog tekst processore (npr *Word For Windows*) rezidentn h vodiča (*Norton Guide*) iii Hiper Teksl (*HyperText*) baza podataka

Trenutno zbirka evakvih izdanje na Sezamu broj oko 70 naslova arhiviane dnžine prako 8 Mb (a obzirom da sa tekstovi dobro pakuju realne dužina daarhiviranih datoteka je nekoi ko puta veca)

Pre izvesnog vremena smo pisali o informacijome koja Microsoft distribuire za konsnike *Windows*a a mogu se naći na Sezamn ("Računari" 92) e ovde cemo se više osvrnuli ne elektronske zbirke informacije namenja ne programerima

Sigurno bi veliki broj vlasn ka PC računara sa Intel i80386 ih jačim procesorom želeo da se bar malo okuša u programiranju na mešinskom nivou Na žalost sem senje lekstova o programiranju 386 u Ročnnanima ovde nije bilo drugih izvore informacije Ali sada se ne Seza mn mogu naći odlični lelesalnu vodiči za programiranje Intel i80386 procesola ! njegovog matematičkog koprocesora Intel i80387 Izvori ove dve knjige (u obl ku tek stualne datoteke) je sam Intel Corporation a sve ih zaista na pravu knjigu – ako hocete možete ja odštampati!

"Knjiga" je podeljena na poglavlja koja vas polako uvode u materiju počev od opisa osnovn h pojmove preko organizacije memorije i memorijskih modela tipo-va podataka procesorskih registara vrsta nstrukcija pa sva do obrade raznih grešaka opisa multitaskinga i mul tiprocesorskog rada Obraduje se i rad u virtuelnom 8086/8086 modu *80286 protected* modn narevno i pu nom 386 režimu Naslov ovih knjiga je "Intel 80386 Programmer s Reference Manual" (odnosno 80387) a pe obumn su zaista pozamašne – uputstvo za 386 procesori sadrži uknpno 900 Kb teksta (21 500 redova) dok za 387 koprocesor ima oko 450 Kb ili skoro 10 000 redova Kade bista sve to odštampali na hartijn standardnog A4 formata dobili bi bio pravu kolijn od nekih 500 strana Ako vas sve ovo nlelnasje pokupite iz INFO direktorijume datoteka i386 a1 j, i386 a01 i i387.zlp

Ako konstite Borlandov *Trnbo Pascal* o nemata "Trnbo Pascal 6 kompletan vodič" Stvena O'Brejno (Stephen O'Brien) izdal kod nas onda ce vam vrlo kon sna bit besplatna (*freeware*) elektronska knjiga "Turbo Pascal Relaisnce čij je autor Ed Mičel (Ed Mitchell) Ova knjige sadrži detaljne referentne informacije vezane za *Turbo Pascal 6 0 Turbo Pascal 7 0* i *Borland Pascal* ali i za *Turbo Vision* za koji ovde nema neke literature osim originalne Borlandove (\ibmpc\pascal\tpr-book.arj – a01)

Jedna od najčešćih tema za razgovor među Clipper programerime je pisanje programa koji treba da rade na nekoj mreži računara (uglavnom *Novell NetWare*) Na programiranje pod mrežom se često gleda kao na tauk iako u principu pravile koji se treba držati nisu previše teška već je problem u nedostatku odgovarajuce litera ture i još više n nedostatku iskustva jat nema svako pristup pravoj mreži da proba kako sve radi n praksi

Dobru osnovu za početak pruža knjiga n elektron skom obliku pod naslovom "Clipper in Network I-ps sug gestions " alias "Network Guide for Clipper programmer" Iako je talanvno malog obima (kada se odštampa više ići na neku brosnu polito se dobije svega oke 25 strana formata A4) početnike ne rezumljiv način poste peno uvodi u svet programiranja mreznih aplikacija n Clipper u Obrađena su osnove deljenog (*shared*) pristu pa zaštita podataka zaključavanje slogova i datoteka problemi sa memorijom podešavanje softversk h i hardverskih resursa na mreži si S obzirom na mali obim knjige normalno je da nje sva obrađeno do detalje a nešto je čak i preskočeno (npr problem štampanje pod mrežom) ali sigurno cete nešto naučiti (\ibmpc\clipper\netguide.zip)

Od ostalih novitela u INFO direktorijumu treba spo menuti još i dva dobra recn ka a za ostale ce se naći meste nekom drugom prilkom **PC-Glossary v5 1** (datoteka **glosry51.***) sadrži objašnjenja raznih mikroračunar skih termina i skracenica (akronima). Objašnjenja su do voljno informativna prosečno oko jedan aklan teksta po pojmu...a snalaženja i pretraživanja je lako

Drugi je rečnik kompjuterskog slenga **Jargon list v2 9 11** Predstavlja spisak sa detaljnim objašnjenjem žargonskih izraza koj se konste među hakerskom i drugom populacijom vezanom za rad na računarima. Zbirka je vrlo velika (1 1 Mb tekste) i ako ste novi na neko računarskoj mrezi vrlo ja konsno imati je pn ruci jer ve-ćinu izraze koji se tu konste n svakodnevnoj komunikaciji nećete naći ni u jednom standeidnom rečniku Nalazi se u \ibmpc\info kalalogu arhive **jar_2911.***

Utility

Za ova In meseca izašla su i In nova verzija McAfee paketa za borbu protiv virusa od koj h su za nas najzanimljivi *Scan* (za proveru ima li virusa ne vašem sistemu) i *Clean* (za njihovo uklanjanje) Prvo se pojavile ver zija 9 15v104 (v103 je preskočena) zatim v105 e n dru goj polovini juna varzije **Scan 9 17v106** koja je još uvek aktuelna

Uz standardnu podršku otkrivanju sve više novih vi rusa značajan novitel od verzije 104 je ukidanje progre me *NetScan* koji je služio za skaniranje mrežn h diskova Frankcije *NetScan*-a su integnsene n standardni *Scan* koji se sad može konstiti za pretraživanje lokalnih i mrežnih diskova a za potpunju zaštitu mreža služi *Net Shield* NLM (*NetWare Loadable Module*) za *Novell Net ware/386* Broj virusa koje otkriva (a većinu i nklanja bez problema) *Scan* paket sada je porastao na 1 453, ili čak 2 149 uključujući njihove modifikacije

Ukoliko vile volite manje i rad sa mišem od stan dardnog konšcenje McAfee anti-virusa iz komandne linije onda će vam morala zgodan bti *shell* program za *Scan & Clean* pod nazivom *The Virus Terminus* Nalazi sa u arhivi **vtec45c.zip** direktorijum je \ibmpc\virus

Jedan od željno očekivanih novitela je prva zvanič-na varzije **ARJ** arhivere poste dugog perioda testenja I pel probnih (beta) međuverzija. Verzija 2 41 i 2 41e koje je poslednje aktnelna je n odnosu na 2 30 donela po-boljšanje pre svege u brzni rada I pouzdanosti a primeću-ju se neki mali dobici n stepenn kompresye Ipak brz na rada ja još uvek daleko od novog ZIP-a I to je jedina (ali vrlo značajna) mana ARJ a (\ibmpc\archiver \arj241e.exe)

Ako vam je teško da čitate redom svu ARJ doku-mentaciju (e ima je zaista dosta) onda pogledajte zva-ničnu *HyperText* knjigu o ARJ u gde je sve oranizovano teko da bez problema možete za čas da nadete opis šnjanja bas onih opcija koje vas interesuju (\ibmpc\ar chiver\arjhlp24.arj)

Tabela 1 – TOP 10 datoteka – jul 1993

	direktorijum	datoteka	opis
1	\IBMPC\VIRUS	scanv106	a07	McAfee SCAN 9 17v106, virus detektor
2	\IBMPC\ARCHIVER	arj241a	exe	ARJ arhiver (v 2.41a)
3	\IBMPC\VIRUS	clean106	a07	McAfee Clean 9.17v106 uklanja viruse koje SCAN nađe
4	\IBMPC\INFO	mnp10	zip	Šta je to MNP10 protokol?
5	\IBMPC\UTILITY	list78a	zip	Pregled datoteka zamene za type (v 7 8a)
6	\IBMPC\WINDOWS	wks_	zip	Windows strayer za YU tastaturu (dikinac)
7	\IBMPC\INFO	maxtor	zip	Tehnički podaci o Maxtor hard d skovima
8.	\IBMPC\UTILITY	ztime14	zip	ZTime v1 4 rezidentni sat na ekranu (dicalec)
9	\IBMPC\WINDOWS	winzip41	zip	WinZip 4 1 shell za rad sa ARJ/ZIP/LZH/ARC arhivama
10.	\RSOFT	r083tsr	zip	Rezidentni program – jedan pravi TSR ("Računari" 93)

BEST BUY

386 SX 33

1950 DM

2Mb expandable to 16Mb
1 44Mb floppy
105Mb (15mS) hard drive
512K SVGA card
SVGA monitor color 14"
DOS 6 0 + windows 3 1

2 godine garancija

BEST BUY

386 DX 40

2300 DM

4Mb expandable to 32Mb
128K cache 1 44Mb floppy
130Mb (12mS) hard drive
1Mb SVGA card
AVGA3 16 7 milion colors
SVGA monitor color 14"
DOS 6 0 + windows 3 1

2 godine garancija

BEST BUY

486 DX 33

2800 DM

4Mb expandable to 32Mb
256K cache 1 44Mb floppy
130Mb (12mS) hard drive
1Mb SVGA card
AVGA3 16 7 milion colors
SVGA monitor color 14"
DOS 6 0 + windows 3 1 + mouse

2 godine garancija

BEST BUY

486 DX 50

3200 DM

4Mb expandable to 32Mb
256K cache 1 44Mb floppy
130Mb (12mS) hard drive
1Mb SVGA card
AVGA3 16 7 milion colors
SVGA monitor color 14"
DOS 6 0 + windows 3 1 + mouse

2 godine garancija

COMPONENTS

BOARD 386 DX 40	330 DM
BOARD 486 DX 33	900 DM
BOARD 486 DX 50	1250 DM
HDD "W DIGITAL" 130Mb	450 DM
HDD "W DIGITAL" 220Mb	600 DM
HDD "W DIGITAL" 255Mb	680 DM
FAX/MODEM CARD INTERNAL	280 DM
1 Mb SIMM	120 DM
SOUND BLASTER CARD	400 DM
MONITOR COLOR 14"	600 DM
MOUSE GENIUS "CLX PROFY"	80 DM
EPSON LX400 YU SET	480 DM
EPSON LQ100 YU SET	600 DM
EPSON LQ170	1400 DM
HELWETT PACKARD llp	2500 DM

MP–biro PREDNOSTI

Visoki kvalitet, najniže cene i mnoge dodatne povoljnosti:

* isporuka opreme na kuću
* ugradnja dodatne opreme
* mogućnost kasnije dogradnje sistema
* sastavljanje računarske opreme prema vašim potrebama
* programska podrška

WHAT PC?

PCDirect

PC

COMPUTER BUYER

NOVELL

MICROSOFT WINDOWS

SOUND BLASTER

MP BIRO

COMPUTER SYSTEMS

BEOGRADSKA 41
TEL 011/341-392
496-351

A sada Vi Combinujte

CT 386SX -105

CPU 386SX/25 MHz
2MB RAM
Floppy 1.2 MB i 1.44 MB
Tvrdi disk 105 MB IDE
SVGA 1024 x 768 / 512 kB
SVGA kolor monitor
slimline kućište mid

CT 386/40 -125

CPU 386DX/40 MHz
4MB RAM cache 128kB
Floppy 1.2 MB i 1.44 MB
Tvrdi disk 165 MB IDE
SVGA 1024 x 768 / 1 MB
SVGA kolor monitor
mini tower kućište mid

CT 486/33 -210

CPU 486DX/33 MHz
4MB RAM cache 256kB
Floppy 1.2 MB i 1.44 MB
Tvrdi disk 210 MB IDE
SVGA 1024 x 768 / 1 MB
SVGA kolor monitor
mini tower kućište mid

CT 486/50 -210

CPU 486DX/50 MHz
4MB RAM cache 256kB
Floppy 1.2 MB i 1.44 MB
Tvrdi disk 210 MB IDE
SVGA 1024 x 768 / 1 MB
SVGA kolor monitor
tower kućište mid

TIM TALENATA

EPSON

FX-1170

EPSON

LQ-1070

HEWLETT PACKARD

Desk Jet 500
ink jet štampač
rezolucija 300dpi
A4 bufer 16 kB
kvalitet ispisa kao
laserski štampač!

HEWLETT PACKARD

HP LJ 4L
laserski štampač
rezolucija 300dpi
1MB memorije
RET tehnologija
YU slova

HEWLETT PACKARD

HP LJ IV
laserski štampač
rezolucija 600dpi
2MB memorije
RET tehnologija
45 fonteva

CASIO

DC-7500
Digital diary

CASIO

MS-470L
14 cifara stoni

CASIO

SF-9300
Digital diary

ComTrad **računari su dugo tražene kombinacija savršenog sastavljanja dalova u kompaktne caline.**

Sada VI pronađite kombinaciju koja Vama odgovarai

Garancija 12 meseci za računare i štampače, 6 meseci za kolkulatore. U cenu štampača uključen Centronix kabl

Combinacija Tražana dugo
Genex apartmani
Vladimira Popovića 6

ComTrad

011 - 222 - 41 - 51
011 - 222 - 26 - 52
011 - 222 - 41 - 39

■ **function KeyPressed: bo**

Vraća **true** ako je u baferu bar jedan znak.

■ **function Lo (ii): byte**

Vraća niži bajt argumenta koji mora biti tipa **integer** ili **word**

■ **procedure LoVideo**

Aktivira ispis standardnim slovima.

■ **function MaxAvail: li**

Vraća dužinu najvećeg slobodnog bloka na **heap**-u

■ **function MemAvail: li**

Vraća ukupan kapacitat slobodne memorije na **heap**-u. Ovaj prostor se mora biti kontinualan dimenzije najvećeg kontinualnog bloka dobijaju se se **MaxAvail**.

■ **function MSDos (var regs:registers)**

Zahteva usluge MS DOS-a.

■ **procedure NoSound**

Isključuje interni zvučnik.

■ **function Ofs(x): wo**

Vraća ofset objekta **x**

■ **function ParamCount: wo**

Turbo Pascal omogućava da se programu prosleđe parametri zadati u okviru komandne linije kojom je startovan. Ova funkcija daje **broj** tih parametara

■ **function ParamStr (wo): st**

Daje parametar broj **wo** iz komandne linije kojom je program startovan.

■ **procedure Randomize**

Priprema generator slučajnih brojeva, koristeći interno vreme kao **seed**.

■ **function ReadKey: ch**

Vraća znak koji je otkucan na tastaturi, ako u baferu nema znaka, čeka se da korisnik nešto otkuca. Specijalni tasteri daju specijalne **scan** kodove.

■ **procedure RunError (wo)**

Prekida izvršavanje programa i generiše izvršnu grešku čiji je kod **wo**. Ova mogućnost se najčešće koristi kod pisanja složenijih programskih paketa čije se komponente povezuju preko 'EXIT' procedura, omogućujući sistemu se tada može upravljati primenom DOS promenljive ErrorLevel.

■ **function Seg (x): wo**

Vraća segmentnu adresu objekta **x**.

■ **procedure SetCBreak (bo)**

Postavlja DOS Ctrl Break prekidač. Ukoliko **bo** ima vrednost **true**, DOS će testirati stanje Ctrl i Break tastera kad god je tražena bilo kakva njegova usluga. **bo=false** nalaže da se testiranje vrši samo kod ulazno-izlaznih operacija.

■ **procedure SetDate (year, month, day, dayweek: word)**

Podešavanje sistemskog kalendara. **Dayweek** ima vrednosti 0 do 6 pri čemu nula označava nedelju. Ako je da tom besmislen biće ignorisan

■ **procedure SetIntVec (by, vector: pointer)**

Postavlja interapt vektor koji odgovara Interaptu **by**. **Vector** se obično zadaje uz pomoć operatora @ koji da je adresu interaptske procedure

■ **procedure SetTime (var hour, min, sec, sec100: word)**

Podešavanje sistemskog vremena.

■ **procedure SetVerify (bo)**

Postavlja DOS Verify prekidač. Ako je **bo=true** svaki upis podataka će biti praćen proverom ispravnosti upisanih podataka.

■ **procedure Sound (wo)**

Aktivira interni generator tona, **wo** određuje frekvenciju.

■ **function SPtr wo**

Vraća trenutnu vrednost registra SP

■ **function SSeg: wo**

Vraca trenutnu vrednost registra SS

■ **procedure SwapVectors**

Zamenjuje kompletnu sonu vektora procesora 80x86 sa SavelutXX zanom samog **system unit**-a. Ideja je da se zamena izvrši pre i posle izvršavanja nekog programa (*Turbo Pascal* procedura **Exec**) kako bi se obezbedilo da svi ranije instalirani rezidentni programi i interapt handleri budu očuvani

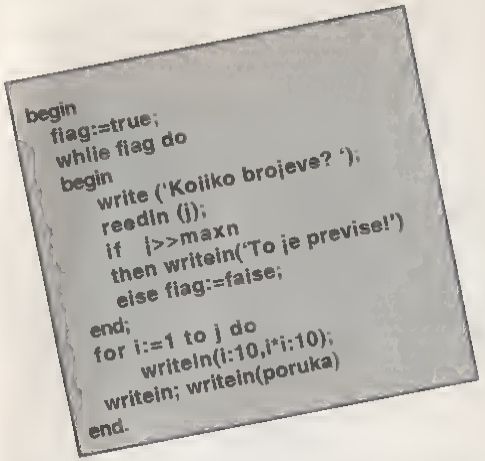
■ **function WhereX: by**

Vraća trenutnu **x** koordinatu kursora.

■ **function WhereY: by**

Vraća trenutnu **y** koordinatu kursora.

Dejan Ristanović



MOJA ŠKOLA PASKALA

SPECIJALAN DODATAK ČASOPISA računari
septembar 1993

1. UVOD

Neko je duhovito primetio da se na može imenovati jezik koji bi bio **najsavršeniji** među programerima, ali da je **drugi** najomiljeniji programski jezik – paskal. I zaista, neki programeri vole tradiciju fortrana, drugi još nestašniji bejzika... treći kompleksniji Ade, četvrti su pristalice Modula 2 – peti (i verovatno najbrojniji programeru ne C-u i take dalje ali je paskal jezik od kojeg je velika većina počela i kojem se mnogo povremeno vraćaju. Teško je i zamisliti programera koji ne poznaje osnove paskala jezika koji po mnogima, predstavlja najbolju moguću ulaznicu u svet programiranje. Zato će i ovaj umetak poslužiti kao svojevrsna ulaznica u svet paskala – pročitajte ga, uprobajte *Turbo Pascal* i progra mirajte na njemu. Znanje koje ćete steći svakako će vam pomoći da razumete logiku svih drugih programskih jezika a možda ćete ga ona i dementovati računicu kojom ovaj pokoli navodivši paskal za svoj najomiljeniji jezik!

Poput većine velikih ideja paskal je dalekosežnim ambicije svog tvorca – prvobitno zamišljen kao jezik za upoznavanje budućih kompjuterskih stručnjaka sa savremenim pristupom programerskom zanatu paskal je danas jedan od jezika koji se koristi za razvoj komercijalnog pa čak i sistemskog softvera. Naravno za razvoj softvera on na koristi baš prvobitna verzija paskala pa čak nije sasvim sličan ni sam duh paskal je zamišljen kao mašinski strogo nezavisan jezik koji će obezbediti da se jednom napisani program bez ikakvih izmena izvršavo na svakom kompjuteru koji je opremljen odgovarajućim kompajlerom pokazale su, međutim, da se na mnogim računarima bez određene doze mašinske zavisnosti pesan jednostavno ne može racionalno ostvariti. Što se PC sveta tiče, dobro standardizovani paskal bi bez sumnje stekao reputaciju jezika koji vredi znati a ne vredi koristiti: da nije bila firma *Borland* čija je *Turbo Pascal* pramda na baš sasvim mnogim se standardi ma, naprečac osvojo milione PC-ovaca širom sveta.

Koje je te nove kvalitete *Turbo Pascal* ponudio? Pre svega, komfor – svaka nova verzija donosila je bremu prevođenja izvornog programa koja je ranije bila jedanstavne nesumalive i koja je dokazalo da se sa jed nim kompajliranim jezikom može raditi onoliko komfor na koliko i sa bejzikom. Sam jezik je dalje, proširen tolike do pedrži rad sa strukturama i datotekama, grafi ku, zvuk i nikratke sve ona bez čega jedan savremeni program ne može da opstane *Turbo Pascal* najzad sjaj ne saradnja sa MS DOS-om što mu otvara brojne perspektive u sferi sistemskog programiranja koje su na drugim mašinama retke povezuje se paskalom – drukčan pristup memoriji pisanje interpt rutina, komuni kacija sa periferijama. Takav *Turbo Pascal* predstavlja najpraktičniju alatku za brz razvoj PC softvera i zato se u svetskim razmerama, koristi kao osnovna alatka za izradu programskih paketa za internu upotrebu u okviru korporacija što je prema mnogim istraživanjima, posao na kojem je angažovana 60-65 posto programerske populacije.

Početkom 1993 godine *Borland* je odlučio da svoj paskal kompajler podeli u dve grane – u jednoj je takozva ni *Borland Pascal* (tekuća verzija ima oznaku 7.0) koji predstavlja profesionalnu alatku za razvoj softvera za DOS i *Windows* okruženje, a u drugoj *Turbo Pascal 7.0* jeftiniji i jednostavniji kompajler koji je namenjen spe cijalizovanoj paskala i programiranju pod DOS-om. To je odela ne bi trebalo da predstavlja ozbiljnu smetnju za pra đenje svog teksta – baviđemo se paskalom kao jez ikom...tako će biti praktično sve što kažemo važiti kako za *Borland Pascal* i *Turbo Pascal* 7.0 take i za neke od ranijih verzije svog kompajlera (posebno 5.5 i 6.0).

Nive izlaganja prilagođen je relativnim početnicima koji su možda imali nekih kontakata sa programiranjem na bejziku fortranu ili nekom drugom jeziku ali koji imaju mala ili ni mala iskustva u programiranju na paskalu. Takav umetak će predstavlja zamenu za originalnu dokumentaciju paketa *Borland Pascal 7.0* - preporučujem da ga knjige držite „pri ruci" jer ćete u njima naći precizna opis razvojne okoline sintakse, sistemskih pro cedura i funkcija i druge stvari bez kojih je teško programirati.

2. PRVI KORACI

Pokušajmo da napišemo neki ulazak u paskalu pomatrajući jedan zaista jednostavan program koji pronalazi aritmetičku sredinu pekoliko unetih brojeva.

```
program prosek (input,output);

{
   Izračunava aritmetičku [illegible]
        N zadatih brojeva.
}

var  i,n :integer;
     a,s :real;

begin
    write ('Koliko će [illegible]
    readln (n);
    s:=0;
    for i:=1 to n do
    begin
       readln (a);
       s:=s+a
    end;
    writeln ('Prosek je: ',s/n:
end.
```

Neke bitne osobine se odmah uočavaju. paskal programi se pišu malim slovima, pri čemu bi možda bilo tačnije reći da *Turbo Pascal* jednostavno ignoriše razlika između velikih i malih slova – programe pišeme onako kako nam odgovara, ali ja koristiću (za sledeće mnogo prijatnijih) malih slova osnova izuzetaka. Neki programeri smatraju da je program lakši za praćenje ako se konsen de piše velikim a promenljiva i funkcija malim slovima, pa ćete u literaturi pronaći i takve paskal programa. Što se samog pravopisa tiče, razlike nema – dobro zapamti te da paskal ni po čemu ne razlikuje velika od malih slova, pa su prema tome promenljive **SUMA**, **suma**, **Suma** ili, recimo **SuMa** ne mogu koristiti za smeštanje različi tih podataka.

Takođe primećujemo da sve naredbe se ako 2 i na poči nju od početka reda kazemo da je program „nazubljen" Nazubljivanja n stvari, nije neophodno ave bi bilo u re du da sma naredbe pasli od prve pozicije u redu ali se o pravom smatra de nvlačenje pojedinih redova doprinosi čitljivosti programa utoliko što vizuelno odvaja njegove delove. Uobičajeno je da se n okviru svakog bloka naredbi uvlače za tri do pet blanko simbola ovaj broj se ponekad smanjuje u cilju uštede prostora na „papirnatom" listingu. Pomoć pri nazubljivanju pruža i sam *Turbo Pascal*-ov editor – kada pritisnemo Enter kursor se vraća na poziciju od koje počinje prethodne naredba ako za-

redu treba još više uvući, pritiskamo Tab

Standardni separator instrukcija u paskalu je simbol ; – ovaj znak morate stavljati između dve instrukcije čak i kada se one nalaze u različitim redovima. U prvim danima navikavanja se na paskal redovno često izostavljeni neophodni separator će će rezultirati različitim porukama o greški koje nije uvek lako identifikovati. Možete se, ipak utešiti time da je uvođenjem separatora omogućeno da se jedna naredba prostire u više redova, što se u praksi pevremeno koristi naročito ako se planira održavanje programa u čekom časopisu – skraćenje „gramatika" može štedi deste prostora.

Svaki program počinje određenim brojem uspehodnih deklaracija – na paskalu nazvata da izgledi gomilu redova pre nego što može stvarno početi da se dogodi. Prva linija, (**program proek (input, output)**) govori računaru da je ono što sledi program o nm ljubavno pismo... određuje ime programa (**proek**) i tvrdi da će program učitavati nekakve podatke (**input**) i ispisivati neke rezultate (**output**) Vredi reći da je ovaj red za aspekta *Turbo Pascal*-a nasvim nepotreban i da će program savršeno raditi i kada ga izostavite, vašim programera i pak, započnje program evom deklaracijom uvođenjem na taj način standarda „pravog" paskala

Druga linija nakog programa je komentar – u paskalu se komentari skrivaju vitičastim zagradama ({ }) što znači da sa komentar može prostirati u više redova 5 ekranom da su prva verzija paskale implementirana na računarima sa ekranskim setom znakove standardima je predviđeno da se početak komentara razvoprevno može obeležiti sa (* o kraj sa *)

Sledi bitke definicije koja je započeta službenom rečju **var** (od variable promenjiva) – svaka promenjive koju želmo koristiti **mora** da bnda definisana na samom početku programe Zatim se navode sve funkcije i procedure i na kraju glavni program koji počinje sa **begin** a završiave se sa **end.** – tačka na **end** je obavezna kaka za standardnim paskalu take i na *Turbo Pascal* u jer kompajleru treba dati do znanja da je radm deo progra ma nevršen i da se eventualni ostatak datoteke slobodno može ignorisati

3. OD TEKSTA DO PROGRAMA I NATRAG

Paskal je jezik koji sa kompajlira – uvarni tekst napisanog programa treba prevesti n ovršni kod, startovati ga i posmatrati rezultate. Ukratko, sam kompajler je toliko brz da ćete jedva i primetiti njegov rad trenutak pasle vašeg zahteva, program je prevoden i ovršivanje počinje!

Ne *Turbo Pascal* u možete programirati na dva načina – u integrisanom okruženju ili pomoću linijskog kompajlera. Integrisano okruženje obuhvaće sve što je potrebno za razvoj – editor sa u.se programa, help se brojne in formacijama, kompajler sa prevođenje, debager sa traženje grešku sve je dostupno na pritisak tastera. Linijski kompajler omogućava da program pišete u svom omiljenom editoru, snimate ga na disk, prevedete i ovršavate – se svaku ispravku treba ponove pozivati editor, što znači da je rad snatno ogrnju Početnicima preporučujemo da sklucaju TURBO (ili ŠP ako koriste *Borland Pascal 7.0*) i take se opredele za integrisano okruženje

Upotreba integrisanog okruženja u određenoj meri zavisi od verzije *Turbo Pascal*-a koju koristite – verzije zaključno sa 5.5 su imale odličan nalik na **WordStar**, dok je u *Turbo Pascal* 6.0 ugrađen znatno moćniji editor koji obezbeđuje red sa više programe istovremeno i razna druge pogodnosti ovaj editor je u verziji 7.0 dalje usavršen. Ovde se nećemo baviti ni jednom konkretnom verzijom *Turbo Pascal* a. pomenućemo važne stvari koje su manje-više identične n svim verzijama, dok se detalje možete prilično upoznati koje ste dobili uz paket.

Veći deo okruženje integrisanog okruženja povezan je programu koji razvrsta – to je editor U vrkn su mnogi koji obezbeđuju razne naloge, među kojima je najvažnija Ruta prevođenja i ovršavanje programa U dnu je (ako ga niste isključili) prozor koji koristi debager i dok su u raznim uglovima dodatna informacije o snađenju pojedinih tastera – najvažnije je da zapamtite da pritisak na F1 daje veoma detaljan **help**

HELP biblioteka

Poput svakog nabuljalog programa današnjice, *Turbo Pascal* je dopunjen veoma kompletnom HELP bibliotekom, koja ukida potrebu se stalnim listanjem npustava – uputstvo naravno i dalje treba pročitati, ali će odgovore na ranoge svakodnevne pitanja pružati upravo HELP pa je veomi važne da u samom početku naučimo da koristimo njegove usluge A korišćenje je zasvim jed

anstavne bede je kursor pomeniran na neku opciju jednog od menija, trebarpritisnuti F i ne skrenu će se pojaviti dodatan opis te opcije oka čitav opis ne može da stane n prozor ostatak pregledomu pritisnime na PgUp i PgDn Pritiskom na Esc napuštamo HELP i vrađamo se opciji koju smo prethodnvili !

Ukoliko se n okviru fffff P prozora pojavi neki intaknuti (na monohrom monitorima inverzni na kolur monitori ma ufarkan) tekst računar nem pruža mogućnost da se raspitamo o neku srednoj temi treba sama postaviti kursor na intaknuti segment i pritisnuti ENTER U okviru novog HELP prozore mogu se naći okosetelji na neka treće oblasta i teko de prilične duhima na pretheđni HELP ekran vrađame se pritiskom na Alt F1

Obezbeđen je, najzad i pregled kompletna HELP biblioteke – dva puta pritiskamo F i i na ekranu se pojavljuju sve teme HELP-e koje sehvaljujući hijerarhijskoj organizaciji stoju u jedan prozor Pomicnirnjem kurazra na neku od tema i pritiskom na ENTER dobijame detaljna ukavatenje iz isobrena oblasti Iz editora nam je, najzad, pruđena mogućnost de dobijame informacije o zanem jeziku posmoćrama kurser na ključnu reč, pritisnemo Ctrl F1 i dtame opis.

Upotreba kompajlera

Kompajler ugrađen u radnu prostor *Turbo Pascal*-a zaseđa nje teške upotrebiti oikuseta Alt X R t program je prevođen i S vremena na vreme se, međutim, ukazuje potreba sa nekim „specijalnim efektima" koje pružanje na pomoć takovanih opcija kompajlera. Opcije ili direktive *Turbo Pascal*-a dele se na prekidačke, parametarske i kondicione.

Prekidačke direktive

Prekidačke direktive uključuje ili isključuju neke od karakteristike kompajlera. Zadaju se u vidu komentara koji počinje znakom $, iza kojeg je smeštena same opcija u vidu jednog slova i parametra + (opcije aktivna); odnesne – (opcije isključena) Svaki prekidač ima neku podrazumevanu vrednost koju ćemo ispasti u daljem tekstu – ako je ipis prekidača ($A+) podrazumevana vrednost je plus (uključene), pri čemu je programeru prušena mogućnost da sa ($A) isključi prekidač.

■ **{$I-} – isvršavanje ulaznih iurasa**

Kada je ova opcija aktivna, svaki logički izras se garantovano izračunava od početka do kraja. Zar može drugačije? Zašto da ne – ako je vrednost promenljiva a 5 izras (a=3) and (b=4) je **false** bez obzira na vrednost promenjivu k što znači da računanja vrednosti drugog dela izraza predstavlja čist gubitak vremena Programer će, međutim, ponekad poželeti da furcira ove izračunavanje kaka bi se potvrdio da je vrednost b dafinisana u trenutku izvršavanja ovo naredbe Vredi dodati da su standardi ISO Pascale po ovom pitanju prilično nedefinisani – tvrdi se jedine da se **standardnom ne može garantovati da će logički izraz uvek biti računat do kraje** – *Turbo Pascal* to po potrebi garantuje ili ne garantuje!

■ **{$D+} – generisanje dobagerskih informacija**

Počevši od verzije 5 5 n *Turbo Pascal* je ugrađen interni dabager u *Borland* preporučuju i naka usavršene dabagere koji alpkivacijo resvoj software (posebno *Turbo Debugger*) Da bi sarađnja sa dabagerom bile snađuća, u prevedeni program treba ugraditi linijske brojeve pojedinih procedure na osnovu kojih se n petrebnom momentu može utvrditi kojoj liniji izvornog programe odgovare neki segment objektnog koda. Ukidanjem ovih linija kul se skraćuje i (neznatna) ubrzava ali je sarađnja sa debagerom onmogućena *Turbo Pascal* čak otkuda i ne ubacuje linije u kojoj je nastupala greška!

■ **{$E+} – emulacija aritmetičkog koprocesora**

Počevši od verzije 5 5, *Turbo Pascal* obezbeđuje emulaciju aritmetičkog koprocesora 80x87 ako u izvorni kod ugrađene direktivu {$N+,E+}, rezultujući kod će na računaru koji ima aritmetički koprocesor raditi jednako kao na računaru koji ga nema pruasitvu koprocesora će, jamo nkrasti rad. Direktiva {$N+,E-} generiše znatno kraće kode oblasta i teko de prilične duhima na pretheđni HELP ekran vrađame se pritiskom na Alt F1

Obezbeđen je, najzad i pregled kompletna HELP biblioteke – dva puta pritiskamo F i i na ekranu se pojavljuju sve teme HELP-e koje sehvaljujući hijerarhijskoj organizaciji stoju u jedan prozor Pomicnirnjem kurazra na neku od tema i pritiskom na ENTER dobijame detaljna ukavatenje iz isobrena oblasti Iz editora nam je, najzad, pruđena mogućnost de dobijame informacije o zanem jeziku posmoćrama kurser na ključnu reč, pritisnemo Ctrl F1 i dtame opis.

■ **{$I+} – kontrola ulazno/izlaznih greški**

Ukoliko pri izvršenju programe dođe do ulazno/izlazne greške (nepostojeća datoteka, zadato oljen pristup prepe dnik allanumerički podatak une en umesto numeričkog), izvršavanja programa se el ustavlja i računar prjevijuje greška Paskal je međutim pogodno da sa izvršavanje nastavi i de greška prijavi sam program u tom slučaju u kod treba ugraditi {$I-} e ostre koristiti funkciju **ioresult** sa način koji će mu ogsatti n jednom od sledećih poglavlje Ove je po svaj prilici najčešće kori šćena prekidačka direktiva.

■ **{$R-} – kontrola opsega**

Pre izvršavanje naredbi poput a[i]:=3 računar treba da proveri da li se vrednost promenjive i nalazi u granicama koja su određene kada je niz dimenzionisan. Ova provera zahteva ugrađnju značajnog koja instrukcija koje predstavljaju i što je sklične mnoge neprijatnije u operavaju prevedeni program po će je *Turbo Pascal* jednostavne preskočiti U taku rasvoje programa je, međutim, pogodno koristiti {$R+} i take osigurati prijavljivanje greške u svim kasfiktnim situacijama.

■ **{$S+} – provera prekoračenja staka**

Ukoliko je program veoma složen i, naročite, koristi mnoge lokalnih argumenata, predviđen prostor na stacku mu ne mora biti dovoljan što znači da će se neka povratna adresa jednostavno izgubiti odgovarajuće naredbe restrukcija RET će tada po prilici stvari izazvati krah sustema Kontrolu preboračenje staka, dakle smemo da uključimo same ako je program detaljno testiran

■ **{$V+} – kontrola stringova**

Prilikom prenošenja parametara u procedure, *Turbo Pascal* zahteva da formalni i stvarni argument budu i dentičke deklarisani što, pored ostalog znači da stringovi moraju biti iste dužine Korišćenjem opcije {$V } omogućavama prenošenju stringove bez provere dužine, što je prilično pogodno ali i nezasgldeno sa konzepcijom paskala kao jezika.

Parametarske direktive

Parametarske direktive, u skladu sa svojim imenom, obezbeđuju neke operacije kojima je neophodan parametar u vidu imena neka datoteke ili bespostatne memorije

■ **{$I ime} – umetanje paskal programa**

Direktiva **Include** obezbeđuje umetanje kompletnog programa iz datoteke ime u tekući program – baš kao da je segment upisan red po red. *Turbo Pascal* podrazumeva da je program upisan u datoteku čija je ekstenzija PAS i koja se nalazi u direktorijumu specifiziranom u okviru menija **Options**.

■ **{$L ime} – umetanje prevedenog programa**

Direktiva **Link** obezbeđuje povezivanje programa pisanih na *Turbo Pascal* u sa programskim pisanim na drugim programskim jezicima kao što su fortran, C ili assembler Standardna ekstenzija datoteke je OBJ a standardni direktorijum n kome se sna traži može da se specifiira u **Options** meniju. Povezivanje *Turbo Pascal*-a sa drugim jezikima, sa šalost, izlazi iz domena ovog uputka.

■ **{$M stack,heapmin,heapmax} – alokacija memorije**

Uz pomoć ove opcije definiše se maksimalni memorijski prostor alociran za neku potrebe *Turbo Pascal*-a. **Stack** označava prostor alocirun sa stak i kreće se između 1024 i 65520 bajta **Heap** je struktura na koju sa smeštaju vrednosti promenljivih i mnogi drugi parametri a kreće se između **heapmin** i **heapmax**; prvi broj mora da bude u intervalu [0 655360] a drugi u intervalu [heapmin, 655360]

Kondicione direktive

Normalna je bilo očekivati da se prevođenjem istog izvornog programa uvek dobija isti objektni kod Ne mora, međutim uvek biti tad take – u fazi testiranja programer će poželeti da ugradi neke segmente koji pramđe rebitni sa stanovišta izvršavanja alakšavaju leciranje grešk ovakve segmente svakako zbacujemo iz finalnog programa, ali se nikada na može garantovati da se nećeme vratiti testiranju! Zbog svega toga *Turbo Pascal* obezbeđuje uslovnu kompilaciju – neki segmenti koda se prevode samo ako su se stekli određeni uslovi

Opšti skok kondicione direktive je {$IFxxx} {$ELSE} {$ENDIF} gde su trima tačkama označeni segmenti paskal programa Ukoliko je uslov xxx ispunjen prevodi se segment između {$IFxxx} i {$ELSE} odnosno, ako {$ELSE} segment ne postoji {$IFxxx} i {$ENDIF} ukoliko uslov xxx nije ispunjen prevodi se segment između {$ELSE} i {$ENDIF}, odnosno ništa ako {$ELSE} sekcija ne postoji

Uslov xxx može da bude **DEF ime**, **NDEF ime** i **OPT preklidač**. **DEF ime** je true ako je simbol ime definisan; **NDEF ime** je **true** ako simbol ime nije definisan, dok je **OPT preklidač** (npr {$IFOPT R+}) true ako je preklidač n odgovarajućem stanju (npr. R aktivou) Simbole definišemo direktivama {$DEFINE ime}, a njihove definicije poništavamo se {$UNDEF ime}

Greške u programima

Niko nije savršen pa čak ni programer koji koristi *Turbo Pascal* – u toku razvoja nekog programa svakako će se pojaviti značajna količina sintaksnih izvršnih i logičkih grešaka koje treba prepoznati i otkloniti. Fenomenalno *Turbo Pascal*-ovo konfliktnim situacijama je veoma logično, što znači da upravljanje greškama ne treba da bude problematično.

Ukoliko se pri prevođenju programa nađe na grešku, kompajler ustupa mesto editoru – kursor je da bi stvar bila još jednostavnija, pozicioniran na problematičnu naredbu dok statusna linija opisuje grešku, sve je spremno za ispravku i novi pritisak na Alt R R. Ne može se, ipako garantovati da se greška nalazi u liniji u kojoj je detektovana – ako je neka promenljiva nedefinisana, moraćete da pomerati kursor do početka programa ili do početka procedure i da tamo dodate odgovarajući red. Večina grešaka se, naoružani, može otkloniti za kratko vreme.

Greška nastala pri izvršavanju programa manifestuje se na identičan način pri čemu je varovatnoće da je kursor stvaran na pogrešnu naredbi nešto manja – moraćete da pretražujete skoro segment programa ili čak nekoliko daljene procedura

Vredi pomenuti i ponašanje sistema pri radu sa modulima. Ukoliko je greška nastupila u TPU datoteci a ne u glavnom programu računar će automatski učitati odgovarajući modul i pozicionirati kursor u problematični red. Pasta se svakvim automatskim učitavanjem nepovratno gubi program upisan u memoriju. *Turbo Pascal* će vam pružiti priliku da ga sačuvate – ako pritisnete Esc, modul neće biti ni učitan, što je prilično zgodan kada znate da greška nije u modulu nego u glavnom programu koji pripremа pogrešne podatke!

Integrisani debager

Debager je alatka potrebna svakom programeru – uz njegovu pomoć izvršavanu „nestašnu" program instrukciju po instrukciju ili proceduru po procedura i pronalaženo greške mnogo brže i efikasnije nego kada ručno analiti proveravi recept u štampanju programa i njegovoj čistaljnoj analizi uz korišćenje višebojnih flomastera.

Najjednostavnija upotreba debagera svodi se na pritiska na F7 i F8 koji izvršavaju program red po red – posle svakog reda možemo pogledati rezultate proveriti vrednost neke promenljive i nastaviti se izvršavanjem. Prva stvar koju treba primetiti je da je samovna jednica izvršavanja programa red a na **instrukcija** – ako se u nekom redu nađe više naredbi, sve će biti izvršene jednim pritiskom na F7 odnosno F8.

U čemu je onda razlika između F7 i F8? Ukoliko je u strukciji na koju je kursor pozicioniran bilo šta osim poziva procedure odnosno aritmetičkog izraza u kome figurišu neka funkcija, ova dva tastera deluju identično. Ukoliko se radi o pozivu procedure ili funkcije pritisak na F7 će izazvati ulazak u proceduru i njeno izvršavanje red po red, pritiskom na F8 čitava procedura se izvršave odjednom i kursor se pozicionira na sledeći red. Smisao ove kompliciranje je u tome što velika korisnika najpre isprevi osnovne procedure a onda prelazi na ispravljanje pozivajućih programa ako se prilikom verovatnoćnom smatrenju da je procedura ispravna, njene izvršavanje naredbu po naredbu predstavlja nepotreban gubitak vremena

Dalje mogućnosti obezbeđuje taster F4 sveri goto cursor Instrukcija koja će biti izvršena po pritisku na F7 ili F8 je uvek obeležena velikim kursorom (inverzno osvetljenje blokom ali ja mali editorski kursor i dalje aktivan – moćemo ga pozicionirati na neki od redova izvornog koda i, pritiskom na F4 zahtevati da se izvrši kompletan segment programa, počevši od tekuće instrukcije pa do kursora. Instrukcija da koje se program izvršave ne može se nalaziti u trenutku aktivnom modulu – primenom opcije **Find procedure** iz **Debug** menija možemo pustiti u neki drugi modul i izvršavati program da nekog tamošnjeg segmenta. Vaniki akcikovi se, jasno ristični utaliks šta se možemo biti sigurni da će program uopšte stići do izabranog mesta.

Pri debagovanju složenijih programa korisno će pasti ti takozvane prekidne tačke (**breakpoints**). Upotreba je veoma jednostavna – pozicioniramo kursor na neku naredbu, izaberemo stavku **Toggle breakpoint** iz **Break/watch** menija i linija će biti obeležena kao prekidna na ekranu je ovaj taksi podeljiva. Ukoliko se u toku debagiranja izvršavanja programa dođe do ove linije, rad se automatski prekida i pruža nam se prilika da proverimo vrednosti kritičnih promenljivih i nastavimo se izvršavanjem red po red

Već smo nekoliko puta pomirjali mogućnost pregleda ili promena vrednosti neke promenljive koja je trenutno smještena se monitou programa i to brzine greški. Za ovakve operacije služben je **Debug** meni i njegove opcija **Evaluate** po pritisku na Ctrl F4 na ekranu se pojavljuje veliki prozor podeljen na tri reda U prvom redu u sredinu ime promenljive ili čak kompletan izraz (prozor, dakle možeme koristiti i kao mali interni SidaKlobi u drugom dokjano njegovu vrednost a pomoću trećeg unošenu novu vrednost vrednost se jasno može dodeliti promenljivoj ali na i izrazu Pošto smo unili zadrtaj u treći prozor i pritisnuli ENTER, vrši se sintaksna analiz se unmenog izraza i proverava da li njegov tip odgovara tipu promenljive čija se vrednost menja ukoliko je sve u redu vrednost se trajno dodeljuje i pruža nam se prilika da nastavimo se izvršavanjem programa.

Detaljnije proćenje vrednosti pojedinih promenljivih omogućuje „posmatrački prozor" (**watch window**) Primenom prve opcije iz **Break/watch** menija postavljamo „špijunsku" na neku promenljivu ili izraz čije će se vrednost stalno ispisivati u posmatračkom prozoru Primena ove opcije donosi mnogo problema vezanih se format prikazivanja podataka i činjenicu da su pojedine promenljive definisane u nekom procedurama, nedefinisane vaz njih a u nekim specijalnim slučajevima čak i „preklopljene" drugim vrednostima. Verujući da su problemi veći od dahtaka, posmatračkim tačkama se ovde neće baviti – upućujemo vas na dokumentaciju *Turbo Pascal*-a.

4. PROMENLJIVE I TIPOVI

Sve promenljive koje se koriste u paskal programima moraju se najpre definisati. Drugi bajnik, fortran ili **Clipper** programeri ništa baš odlučivani tim zahtevom ali se na smetra jednim od stožera modernog programiranja – videćete da će vas kompajler baš zahvalnjući definicijama upozoriti na svake pogreške otkucane ime promenljive ili na svaki pokušaj da se ne neka mesta upiše podatak pogrešnog tipa, što će znatno smanjiti broj greški u programima i samim tim, povećati njihovu pouzdanost. Sasatra se, osim toga, da će deklarisanje promenljivih „unterait" programera da zapre na papiru skicira program što blagotvorno deluje na njegov dizajn. Ime promenljive je niz slova i brojeva (obavezno se počinje slovom, dok interpunkcijski znaci u imenu nisu dopušteni) gotove proizvoljna dužina – večina paskal kompajlera ograničava ima na 32 ili 64 znaka (*Turbo Pascal* na 63), ali je to daleko više nego što će vam u praksi biti potrebno. Savetujemo da uvek koristo dovoljno duga i deskriptivna imena promenljivih – to da vam mnogo olakšati snalaženja u programu. Vredi znati da broj slova u imenu promenljive, kao i komentari, nemaju **baš nikakvog** uticaja na dužinu ili vreme izvršavanja pravednog programa!

Imena promenljivih su smeju poćinjati takozvanim rezervisanim rečima ty imenima raznih komandi paskala. Spisak tih reči dat je na slici 4 1 – ukoliko neku od njih zaboravite kao ime promenljive, sva je prilika da će vas kompajler podsetiti nekom čudnom porukom! Radi veće sigurnosti kompajlerskog obrazovanja reči date da je ferirnan jedan od vrlo retkih programskih jezika koji ne razlikuju velika i mala reči. Imena se promenljive mogu zvati DO, WRITE_READ i slične

absolute	else	inline	procedure	with
and	end	interface	program	xor
array	external	interrupt	record	then
begin	file	label	repeat	type
case	for	mod	set	until
const	forward	nil	shl	untill
constructor	function	not	shr	uses
destructor	goto	object	string	var
div	if	of	then	while
do	implementation	or	to	
downto	in	packed	virtual	

slika 4.1.

Slika 4 2 prikazuje nekoliko korektnih i nekoliko nekorektnih imena paskal promenljivih – najlakše se vidi na primerima.

Korektna imena	Nekorektna imena
a	7dana
x650161	x650-161
UcitavanjeVrednosti	ucitavanje vrednosti
xunuodAOdaZ9	for
LajXIV	pasalnaj_programa

slika 4.2

Obratite pažnju na poslednju prenos živu se spiske programih koju sama nazvali **ponekad_pogrešna**. sam slova, ima sve promenljive obuhvata ! donju crtu takozvani **underscore** (_) S obzirom da donja crta nije ni slovo ni cifra, standardni paskal sa dopuška korišćenje ovakvog imena, ali su autori 90% kompajlera odlučili da se ogluše o standard u cilju povećanja čitljivosti programa, tako je **underscore** postao neizbežan elemenat paskala I *Turbo Pascal* podržava ovu ekstenzaju, pa se danje crtice u praksi često koriste. Naki programeri više vole da se pogrešni velikim slovima – umesto **prvi_nis_brojeva** nazvaće promenljivu **PrviNisBrojeva**. Ovakvo imenovanje zahtava određenu disciplinu, pošto je što se kompajlera tiče, **PrviNisBrojeva** iste što i **prvinisbrojeva**, dok su **prvi_nis_brojeva** i **prvi nisbrojeva** i te kaka razlikuju

U definiciji se–sam imena, obavezno navodi i tip promenljive – se razliku od bejzika ili fortrana, taj tip nije povezan se imenom već se svakom imenu može pridružiti bilo koji tip. Najčešće deta koristi skalarne promenljive, kao mogu da budu celobrojne, racionalne, logičke ili alfanumeričke

Skalarne promenljive

Celobrojna promenljiva se deklariše korišćenjem reči integer i može da „zapamti" isključivo ono broj između **maxint** i **-maxint** Vrednost **maxint** zavisi od implementacije paskala, ali obično iznosi 2147483647 ili na *Turbo Pascal* u 32767 *Turbo Pascal* je što se celobrojnih promenljivih tiče, ponudio i razne ekstenzije. Tu su, pre svega, promenljive tipa **byte** (0 255), **shortint** ((128 127)), word ((0 65535)) i najpad **longint** ((2147483648 .2147483647)), tip koji šeta verovatno najčešće koristiti, jer je raspon koji dopušta prenutno gotovo uvek što su u realnosti javlje (da nije ove mašinje iz redonice hiuma izlaznih ana **golovo** svuma)

Racionalne promenljive koriste se sa „pamćenje" vrednosti koje nisu cele, a takode se vesma velike ili veoma male brojeve Standardni paskal se definisanja ovih promenljivih koristi tip real vhog tog naziva se to promenljive šeste nazivaju I realnim brojevima, što nije pogrešno ali nije ni napreciznije Tip **real** postoji i na *Turbo Pascal*-u i koristi se na podatke iz opsega [2 *10 $^{-39}$ 1 7*10^{38}] se 11-12 tačnih cifara Za mnoge primene takav opseg je suvvma prihvatljiv ali da saku pokuleti vedu preciznost. Njima preporučujem tipove **double** ((9 0*10^{-324}, 1 7*10^{308})) se 15-16 tačnih cifarni i **extended** ili 3*10^{-4951}, 1 1*10^{4932}) sa 19-20 tačnih cifarai dak će se nai koji šteda memoriju zadovoljiti tipom **single** (1.5*10^{-45}, 3 4*10^{38}) se 7-8 tačnih cifara) Treba, međutim znati da su tipovi single, double i **extended** namenjeni pre svega korisnicima koji su se opremili aritmetičkim koprocesorom ili imaju računar zasnovan na mikroprocesoru 80486 ili na **Pentium** u. Oni koji nemaju koprocesor treba da koriste tip **real** ili da, počevši od *Turbo Pascal*-a 7 0, na početak programa stavne ištve {$E+}. Čime aktiviraju emulaciju – na raspolaganju će im biti svi pomenuti tipovi podataka, ali će red biti nešto sporiji.

Logička promenljiva se, u čast Džordža Bula (George **Boole** 1919 1864 tvorac Bulove algebre) zovu Bulova i deklariše se **boolean** iako je kompajleru zasvim svejedno, ubičajeno je da ova reč započne velikim slovom. Logička promenljiva može da ima samo vrednost true ili **false** pri čemu je standardnim paskala utvrđena da je **false**<**true** – prakticno je false zamena za broj 0 a true za broj 1

Alfanumerička promenljiva se deklariše se **char** i „pamti" samo jedan ASCII znak. Ovo ošte prestavlja veliko kamk nazad u odnosu na fortran 77 ili bejzik, pa sa konstruktori *Turbo Pascal* a uveli i tip **string**, koji se, sa ponašanja kaglika, trećira na zasvim prirodan način – promenljivoj dodeljujemo proizvoljan tekst čija dužina na nam da prede 255 znakova a ostim taj tekst slobodno analiziramo, štampamo i obradujemo na način koji će biti opisan u jednom od sledećih poglavlja String promenljive se deklarišu se **string**[n] gde je n neoznačen broj koji definiše maksimalnu dužinu stringa, ako se izostavi, podrazumeva se 255

```
program tablica (input,output)}
const  max = 1000,
poruka = 'Kraj reda',

var    flag   Boolean,
       zbir   real,
       ulaz   string[28],
       i j    integer,

begin
   flag :=true,
   while flag do
   begin
      write ( Ko[redacted]
      readln (j)
      if   j>max
      then write[redacted]je p[redacted]
      else flag := false
   end,
   for i:=1 to j do
```

```
    writeln(i:10,i*i:20);
  writeln. writeln(poruka)
end.
```

slika 4.3

Slika 4.3 je dobra prilika da utvrdimo ono što smo naučili. Složena reč **program**, kao što znamo označava početak programa koji će uzimati podatke sa tastature (**input**) i ispisivati neke rezultate (**output**). Vidimo je primetili tačku i zarez na kraju većina redova te je separator instrukcija u paskalu, kojima ćemo naše docnije posvetiti dosta pažnje.

Prva sekcija programa obuhvata definicije konstanti. Zbog čega su ona uvedena? Pišete na primer program koji sortira brojeve i odlučite da će ih brojeva biti najviše 1000. Posle izvesnog vremena, zaključujete da brojeva ponekad ima i više, pa odlučite da povećate maksimalnu dimenziju na 2000. Nema druge nego da pretražite ceo program i zamenite 1000 sa 2000- pazite, ipak, da vam ne promaknе ona naredba tipa **for i:=1 to 999 do**. Alternativa je uvođenje promenljive koju ćete, na primer, nazvati **zauzeto** i koju ćete kroz čitav program koristiti kao zamenu za broj 1000- docnije povećanje ima se svesti na promenu naredbe **zauzeto:=1000**. Tvrdnje paskala je upravo zaključke da **maxn** u ovom slučaju nije promenljiva, pošto se njena vrednost na mnoge u toku izvršavanja programa **maxn** je zapravo konstanta! Kada naučite da programirate na paskalu, svaki ćete program započinjati uvođenjem konstanti tj parametara programa koje ima smisla menjati u raznim situacijama- posle svake promene neke od konstanti, program se jasno mora ponovo kompajlirati.

Konstanta se u standardnom paskalu ne deklariše već tu se odmah dodeljuju vrednosti- stvar je zaista ingenia, pošto je konstanta zamena za broj, pa se ona gubi u toku kompilacije- svaka naredba for i:=1 to **maxn** 1 **do**, u našem primeru, biti prevedena kao da smo napisali **for** i:=1 to 999 do.

Sledeći odsjak programa sa slike 4.3 započinje sa **var**- što znači da deklarišemo sekciju promenljivih. Iza svakog imena sledi dvotačka i reč koja opisuje promenljivu (za sada poznajemo samo deklaracije **integer, real, Boolean, char** i **string**), dok se svaki red završava tačkom i zarezom. U poslednjem ime redu definisali smo jednu promenljivu i j, razdvojivši ih zarezom- nema potrebe da vile puta pišemo reč **integer**, pramda ona za to ni smetala.

Ostatak programa predstavlja izvršni deo kojim ćemo se bavit tek u sledećem poglavlju, varajmo, ipak, da ćete lako zaključiti da program ispisuje tabelu kvadrata uzastopnih celih brojeva.

Programe sa slike 4.3 pokazuje da se svaka promenljiva mora definisati, ali i da se ona definisana promenljiva ne moraju koristiti - promenljiva **ukaz** na figurde u izvršnom delu programa. Niko naravno, neće uvesti promenljive koje mu nisu potrebne ali se povremeno desi da se pri prepravci programa ukloni segment koda koji radi sa nekom promenljivom, a istovremeno se na obrisi i njena definicija. To kompajleru ne smeta, ali pak nije preporučljivo – prostor predviđen za pamćenje te „fantomske" promenljive napresto propada!

Tipovi po želji

Jedna od zanimljivih osobina paskala je da nam ograničenost na samo nekoliko tipova podataka, omogućava da uvodimo i sopstvene tipove, što će pokazati i primer 4 4.

```
program test (input,output);

const pi = 3.14159;

type  pozitivan = 1  maxint;
      mifra = 1..9;
      valuta = (franak, marka, dolar, funta,
      boje = (crna, žuta, zelena, bela),
      procesor = (18086, 186286  I80386,
I80486, Pentium);

var   brojac   valuta,
      i,j      pozitivan;
      računar  procesor;
      tanst.   boje:
```

slika 4.4

Imenu svakoj konstanti (**const**) i promenljivih (**var**) smo ubacili ravno sekciju koja je započeta sa **type** radi se o deklarisanju tipova. Pogledajmo najpre tip **pozitivan** koji može da uzima bilo kakvu celobrojnu vrednost između 1 i najvećeg sekog broja koji računar poznaje (npr 32767) Kakav je smisao uvođenja svega ovga- kada smo jednostavno mogli da koristimo tip **integer**? Ukoliko naka veličina pa svoje prirodi mora da ludi pozitivan broj- njen naka znak u „negativna vrda" izvračava da av program nešto u najredivalnoj situaciji koju je najlakše signalizirati korisnik će tada znati da su rezultati verovatno pogrešni. Pažkal prevodilac će, dakle, naredbi u resultujući mašinski program narediti koji proveravaju da li su vrednosti promenljivih naksa u predviđenim limitima i ispisuju odgovarajuće poruke u kmiflibilnim situacijima, ova će nam poruka vesma pomoći pri testiranju programa i uklanjanju bagova- Nema naravno potrebe da se ograničavamo na velika opsega- ukoliko naki brojač nacrta da se nalazi između 1 i 9 (ili 30 je akuratno- i konstante **maxn**) zapisaćemo naše pojmi **type cifra =**
1..9

Daleko je zanimljivije linija koja opisuje tip **procesor**- iza znaka jednakosti smo u obličnim zagradama napisali imena nekakvih **brojeva** nabrajanjem imena su razdvajena zarezima, a ceo sadržaj da svaki ima mora počinjati slovom pisali smo na primer **I8086** a ne **8086**. Promenljivoj **računar** je nakalbu linija dozvoljo dodijeljen tip **procesor**. Ako znači u deniu u programu nači da napišemo nešto poput **računer:=I80286**- Vele je bitno da primetimo da **računar** nije alfanumerička promenljiva da smo napisali **računar:='Pentium'** ra dilo bi se o prostom dodjeljivanju teksta. U paskalu, sa druge strana, govorimo o tipovima koji su mnogim razvanprevu se **integer** ili **Boolean** imaju nikakve analogije da napišemo **for računar:=18086 to Pentium do** po će brojač u petlji redom uzimati vrednosti imena svih procesora.

Naredba **for računar:=18086 to i'entium do** nas ub jeđ nešemu **vrlo** je bitna kojim smo redom navell mi kroprocesore Sledih smo hronologiju [vajavljivanja što znači da će u petlji koju pominjemo brojač dobiti najpre vrednost **18086** zatim **180286** pa tek na kraju **Pentium** Da smo sa druge strana napisali **for računar:=Pentium to I80286 do** sadržaj petlje ne bi bio izvršan ni jedan jedini put, pošto se potrasumjeva da je „broj" **Pentium** već od „broja" **180286** Slika 4 4 opisuje još nekoliko primera kao što su **boje** i **valute** – nema nikakvih prepreka da sami definišete baš one tipove promenljivih koje su potrebne da bista rešili problem koji vam je postavljen. Pre nego što se previše oduševite mogućnošću definisanja novih tipova podataka, pomenućemo jedno ograničenja koje bi bilo pristajalo nekom od sledećih poglavlja. ako modete slobodno da operišete sa promenljivima raznih tipova njikave unošnje i štant perija je praktično nemoguće argumenti procedura **read** i **write** koje predstavljaju vezu programa sa „spoljnim svetom" mogu da budu samo promenljive tipa **integer, real, Boolean, char** ili **string**, što znači da bi **write (računar)** isasvale grešku pri prevođenju.

Nizovi i matrice

Jaka paskal nudi desta struktura podataka koje ćemo tak upoznati, nizovi i matrice su uvek najpopularniji. Pogledajmo, dakle sliku 4.5.

```
program matrice (input,output);

const maxn = 80.

type  indeks  = 0  maxn.
      matrica = array [indeks] of real;
      valuta  (franak. marka, dolar
funta);

var   podatak     array [0  100) of integer;
      promenan    array [ 199  +80] of real,
      ae          array [indeks] of real.
      bb          matrica,
      nc          array [Boolean] of integer,
      kurs        array [valuta] of longint,
      sex_kurs    longint,
      sex_valut   valuta.
      brojač:     valuta;
```

slika 4 5.

Prve definicija je najjednostavnija. uvek smo nis želih brojeva **podatak** čiji indeksi uzimaju vrednosti od 0 do 100 Za razliku od bejzika i fortrana, u paskalu se mogu definisati **obe granice niza** se podrazumeve sa da indeksi počinju od nule ili od jedan Navedenje dodatnog podatka pomaže fleksibilnosti ako se nekada indeksi po svojoj prirodi budu nalazili između 10ti i +100 napisaćemo **array** [-100..100]

Treće definicija sa slike 4 5 predstavlja varijaciju na temu unesete da dimenzionišući niz u srednjim zagrada ma navedena opseg indeksa, definisali smo taj opseg kao tip **indeks** a zatim napisali **array [indeks] of** real Da li je na svaj način naše dobijeno? Ustvedi ome, pre svega, malo pisanje i razmišljanja ukoliko u zaram potprogramima budemo morali da dimenzionišemo matrice ustog tipa. Učinili smo, što je možda mnogo važnije nad programom ostaje čitljivim Dalje unapređenja čitljivosti nudi definicije matrice **bb** tip **matrica** je definisan kao **array [indeks] of real**, a zatim smo napisali **bb matrica** što je, se stanovišta prevođenja programa ekvivalentno sa **bb: array [0 .80] of real,**

Definicija **nc array [Boolean] of integer** je dobra prilika da proverimo značenja Znamo da logička (ili Buolova) promenljiva mogu da imaju vrednosti **true** i **false**, što znači da paskal sadrži implicitnu deklaraciju **type Boolean = (false, true)** Matrice **nc** će prema tome, imati dve elemenata. **nc[false]** i **nc[true]** pa čemo im se taka obraćati u izvršnom delu programa. Sličnc tome bismo mogli da napišemo **ae array [integer] of byte** – dimenzionalnost niz X, čiji se indeksa nalaze između **-maxint** i **+maxint**, zamanjije vaćeg PC-ja je dovoljan da „pojnese" ovaj niz, ali se pojavljuje problem zbog najveće strukture koju *Turbo Pascal* prihvata. Nis je **array [char] of integer** da navedemo još jedan primer bi se sastojao od elemenata y['a'] y['b'] y['z'] Ne smeju se koristiti nizovi čiji su indeksi racionalni brojevi **array [real] of integer** je nepriknatljiva definicija.

Već naslućujete da pri dimenzionisanju matrica nemo ograničeni onovnim tipovima pogledajmo na primer, matrici **kurs** opisanu sa **kurs: array [valuta] of longint**. U izvršnom delu programe možemo napisati **kurs[marka]:=00000650** ili čak napisati program sa slike 4 6, koji određuju valutu čiji je kurs najmanje najviše – pretstite koliko je program postao „čitljiviji" uvođenjem sopstvenih tipova.

```
max_kurs:=0.
for brojac =franak to funta do
    if kvra[brojač] > max_kurs tbo
    begin
        max_kurs:=kurs[brojač],
        max_valut :=brojač
    end.
```

slika 4.6.

Ostalo je još da pomenemo višedimenzionalne matrice, bolje su u paskalu vesma fleksibilne nelose Pogledajmo, dakle sliku 4 7

```
const maxx=20.
      maxy=50.
type  godine   1980  1995.
      mesec  =
(jan,feb,mar,apr maj,jun,jul,avg,sep.okt.nov,dec),

      Dskart = 5  5.
var  x  array [0  maxx, 0  maxy] of real.
       y  array [0  maxx] af array (0  maxy]
of real.
       z  array [1 10,  8  9] of integer,
       prosečni_kurs  array [valute. godine,
mesec] of longint,
       kesardinate  array [Dskart, Dskart.
Dskurt] of real.
```

slika 4.7

Prvi red zvuči savim obično **x: array [0..maxx, 0..maxy] of real** definiše dvodimenzionalnu matricu čiji se prvi indeks kći od 0-50 a drugi od 0-20 Drugi red ja nte to samo malo drugačije **y array [0 .maxx] of array [0 maxy] of real** – baš zvuči komplikovano! Matricu obično zamišljamo kao pravougaonu strukturu od **maxx*maxy** ćelija, svaka ćelija sadrži po jedan, za primer racionalan, broj Matricu, međutim, možemo da zamislimo i kao niz od **maxx+1** kolona (Zašto maxx+1? Zato što počinjemo od nule!) pri čemu se svaka kolona sastoji od **maxy+1** racionalnih brojeva. Upravo u tome i jeste smisao definicije x je niz čiji se indeksi kreću od 0 do **maxx**. Šta su elementi tega niza? Ta mini brojevi nego sam nizovi; indeksi svakog od ovih nizova idu od 0 do **maxy** a elementi niza su racionalni brojevi. Ne sli čan bismo način mogli da uvedemo i trodimenzione setvorodimenzionalne i ostale nizove, ali bi nam prilično kucanja reči array svakako izmorivale – sreća je i omogućava normalno navođenje indeksa u uvodnim sagradi.

Nema posebne potrebe da svi mdaki višedimenzionalne matrice budu u istom opsegu pogledajmo na primer. matricu Z se slike 4 7 – prvi indeks ide od 1 do 10 a drugi od 8 do +6 Mnogo je, međutim zanimljivija matrica **prosečni_kurs**, koja „pamti" prosečne kurseve raznih valuta u pojedinim mesecima određenih godina. U izvršnom delu programa može, dakle, da se napiše naredba koja će glasiti **prosečni_kurs[marka, 1982, avg]:=00000000** i ta će konstrukcija biti jasna čak i nekoma ko nema pojma o programiranju Još nekoliko primera nizova se slike 4 7 protumačite sami

Pomenimo i takozvane **pakovane** matrice, koje se obično opisuju na primeru ormara sličnu možete da nagurate u mali orman ili raspredete u veliki Ako se odlučite za veliki orman, odeća će biti bolje sređena pa četa brže pronalaziti ono što vam je potrebno ali će zato zauzimati više prostora (većina je fioka poluprazna) i biti teža za premeštanja Mali orman podrazumeva nagurani odeću koja se teške pretražuje, ali zato zauzima

manja prostora. Najčešće ćete raditi sa običnim matricama i napisima koje dimenzionišete na način koji sme da smeta upecanah. Ukoliko međutim koristite velike matrici, možete da ih deklarišete kao **packed array** i tako u znatnoj meri uštedite memoriju upotrebljavajući red. *Turbo Pascal* formalno prihvata službenu reč **packed**, ali ne pravi nikakvu razliku između pakovanih i nepakovanih matrica.

Pakovane matrice su jedna od retkih standardnih pretpostavki koje *Turbo Pascal* ne podržava, pomenuli smo samah i zašto njegovu ekstenziju. Na primer, sekcija **const** iz slika 4.8

```
program deskrip (input,output);

type  jezici  =  (fortran,
                  bejzik,
                  ...);

var   brojač:   jezici;
      najveći:  integer;

const deskripcija: array [jezici] of integer
                   =(30,70,100,25,100,100);

begin
  najveći :=0;
  for brojač := fortran to pascal do
      if    deskriptiv[brojač] > najveći
      then najveći :=deskriptiv[brojač];
  writeln (najveći)
end
```

slika 4.8.

Standardni paskal obezbeđuje samo skalarne konstante, dok su u *Turbo Pascal* u konstante zapravo približne promenljivima čije vrednosti da mogu da budu i nizovi odnosno matrice - podatak o diskriptivnosti pojedinih programskih jezika je dodeljen odgovarajućim konstantama u okviru **const** sekcije na način koji vesma podseća na DATA listu u fortranu - drugi nivo dodata vrednosti u glavnom programu obavljen je direktno u okviru **const** sekcije

5. KONTROLNE STRUKTURE

Valjda svi programski jezici na svetu zahtevaju da program podelite na dva dela: deklaracija i izvršne naredbe. Ako smetnete da au deklaracije dosadnoj deci, počnite da se radujete: završili smo se njima (bar privremeno). Da vidimo dakle, kako se pišu izvršni programi.

Iza deklaracije konstanti, tipova, promenljivih i labela (o kojima ćemo govoriti kada povezanost inkrementalnog naredbu goto) sledi procedura i funkcije. Svaka procedura ili funkcija je samu za sebe mali paskal program što znači da ima zaglavlje argumente, deklaracije, definicija svojih funkcija i, naravno izvršni deo. Procedure su, ipak, toliko važna stvar da ćemo im posvetiti čitava poglavlja, ze sada ih preskačemo i prelazimo na glavni program.

U paskalu postoje jedna, na prvi pogled, vesma čudna konvencija. Iako koja procedura, funkcija, petlja ili neka druga struktura uključujući i sam glavni program se sastoji od **jedne jedine naredbe**. Jedna naredba je, jasno zapravo nedovoljna da bi se obavilo bilo šta ozbiljno, pa se umesto nka izvršna rečenica koristi konstrukcija **begin ... end**. Kada paskal kompajler analizira ovaku strukturu i očekuje izvršnu naredbu pronađeno **begin** će izazvati izvršavanje svih naredbi do odgovarajuće naredbe **end**. Program se slika 5 1 sastoji se od nekolike izvršnih naredbi koje su smeštene između **begin** i **end** i međusobno razdvojene tačkom i zarezom. Ovako poređane izvršne naredbe predstavljaju strukturu koje se nimale često naziva **sekvenca**.

```
begin
  i :=0.
  write ( 'Unesi n ');
  readln (n).
  n:=n+i
end.
```

slika 5.1.

Osim sekvence, strukturirane programiranje zahteva ispitivanje i petlju. Ispitivanje se realizuje pomoću konstrukcije **if** – **then** .. **else** ukoliko logički izraz na **if** ima vrednost **true** biće izvršena jedina naredba koja se nalazi iza **then**, dok će u slučaju da izraz ima vrednost **false**: biti izvršena jedina naredba iza **else** (ako sekcija, jasno može i da se izostavi) Umesto jedne naredbe možete da napišete reč **begin** i, iza nje, čitavu sekvencu naredbi (u kojoj smeju da se nalaze i nova ispitivanje) koja se izvršava se **end** „Ponašanje" strukture **if** detaljnije ćemo proučiti na primeru 5 2

```
begin
  if a > 3 then writeln ('A veće od 3 ');
  if b > 3 then
  begin
    writeln ( 8 veće od 3');
  end,
  if c > 3 then
  begin
    s :=c*c;
  end
  else
  begin
    s:=
  end,
  if c > 1
  begin
    writeln
    if x>3
    begin
      p :=x*x+c*c,
    end,
    i :=i+1
  end
  else
  begin
    y :=0. (8)
    if x>3 then
    begin
      p :=x*x e*e; writeln('p=',p)
    end
  end
```

slika 5.2

Prva **if** naredba je sazvan jednostavna: ako je vrednost promenljive A veća od tri, ispisuje se odgovarajući tekst i prelazi na sledeću naredbu B obzirom da smo iepuava nje skavili jednim jedinim **write** par **begin** – **end** s am nije bio potreban mada ne bi smetao da smo ga, kao u drugoj **if** naredbi, napisali Treća **if** naredba je osim komplikovanija. sastoji se iz **then** i **else** sekcije pri čemu svaka od svih sekcija ima po nekoliko naredbi „ugnježdenih" između **begin** i **end**. Četvrti **if** zapsad u svojim sekcijama ima još je jednu **if** naredbu, što nas dovodi do vesma važnog problema gde sve treba staviti separator ';' Smesna li de ga stavljamo sa svake naredbe, nadzirući se da ga paskal ignorisati separatore koji nisu nisu potrebni? Nikako: svaki ustanovni paskala osnuje jednu fatalnu početnu grešku koje može da dovede mnogo glavobolje Pogledajmo naredbu

```
if a=b then. write ("Jednaki su");
```

Simbol '.' koji sme stakli u prethodnoj naredbi je kao što smo rekli, **separator** instrukcija On je stavljen iza **then** i samim tim predstavlja kraj čitave **if**.. **then**...**else** konstrukcije Uzaliko je A jednako B biće izvršeo ceo te **konstrukcija** koji se nalazi iza **then**, što znači da neće biti izvršeno ništa računar će preći na sledeću naredbu (**write**), pa će se on na ekranu pojaviti šetnu tekst 'Jednaki su' rekla bi se da je sve u redu Pretpostavimo, međutim da je A različito od B pošto **else** sekcije se postavi računar će izvršiti prvu sledeću naredbu tj opet **write** na ekranu će se pojaviti tekst 'Jednaki su' iako se brojevi različiti?

Bacimo još jedan pogled se program 5.2 If naredba koju smo obeležili komentarom A line svoj **begin** – **end** koji se završavaju simbolom '.', dok su odgovarajući **if** (obeležan se B) u **else** sekciji se završava svim znakom. Nije štamparska greška. naredbe A i B su ravnopravne utoliko što su na oke elementi **then** odnosno **else** strukture, ali nisu ravnopravne utoliko što je B poslednja naredba strukture dok se iza A, u sekvence, nalazi još i naredba i:=i+1 zato je sauvim prirodno da se iza A pi še tačka i zarez dok ovaj simbol iza B ne bi smetao ali nije neophodan. Verovatno vam u ovom trenutku koriščenje separatora izgleda vesma komplikovano, ali verujeno da ćete se na njih ubrzo navići

Petlje

Paskal podržave tri vrste petlji **while** .. **do**, **repeat** ... **until** i **for** .. **do** Prva petlja se smatra najuniverzalnijom jer iza **while** se može neki logički uslov a iza do jedna naredba (ili **begin** – **end**) koja se izvršava sve **dok je** uslov ispunjen. Pogledajmo kako ova petlja, na slici 5 3, izračunava faktorijel celog broja.

```
program faktorijel (input,output).
const max = 10,
var   i,n,y  longint,
begin
   write  ('n=')
   read
   y :=1,
   while
   begin
      y :=i*
      i :=i+
   end.
   writeln
end
```

slika 5.3.

Iako se još nismo bavili naredbama za učitavanje, svaki će pogodate da se **readln**() zahtevamo unošenje nekog celog broja se tastature program će računati faktorijel ovaga broja y No početku promenljivoj y dodeljujemo vrednost 1 i postavljamo brojač i na istu vrednost, treba da računamo 1*2*3* ^N Petlja se sastoji od dva naredbe (y:=y*i i i:=i+1) koje treba ponavljati sve dok je i manje ili jednako n, pa ukoli i naredba **while** i<=n do o obirom da petlja sadrži dva naredbe neophodna je i **begin** – **end** konstrukcija. Sve je, dakle, jasno ali i pak varijanta da primer 5.3 ne može da proče bez par korekcija.

Šta se događa ako je n=0? S obzirom da je i=1, može biti i i<=n te računar uopšte ne ulazi u petlju – ona će biti preskočena, pa će biti upisane da je vrednost faktorijela 1, što je samvim ispravna. Sami zaključite da li smo mogli da napečensa se I.–2 i take mikrodkopsku ubrzanu program

```
y =1; i:=1
repeat
   y:=y*i;
   i:=i+1
until i>n
```

slika 5.4.

Slika 5 4 prikazuje **repeat** – **until** petlju računar će izvršavati naredbe između **repeat** i **until** sve dok uslov iza **until** ne bude ispunjen (naglasavamo razliku **while** – **do** se izvršave dok je uslov ispunjen a **repeat** – **until** dok uslov nije ispunjen) S obzirom da reči **repeat** i **until** na neki način uokviruju sekvencu, ne moramo da koristimo **begin** – **end** strukturu, prema da ona, jasno, neće biti na smetnji.

Da li možemo da (makrodopski) ubrzamo program 5.4 take što ćemo početi od i =2? Ne šalost, na možemo se n=0 računar će ući u petlju, izračunati y:=y*2 uvećati i, i pošto proveri da je i veće od n, izaći iz petlje, što znači da će 0! biti dva a ne jedan Sadržaj **repeat** – until petlje se, dakle izvršava bar jednom dok se naredba za **while** – do ne mora uopšte izvršiti zato **repeat until** uvemo „petlju se izlazom na dnu", a **while** – do „petlja se izlazom na početku" Za računanje faktorijela je **while** – do petlja nešto pogodnija, ali to se može zahtevati „brojanje unazad" sintaksa je tada **for brojač.=početna donnte krajnja do** Kao što smo vešeli, mogu se koristiti i promenljive čiji sme tip samu definisali na primer **for računar:=18036 to Pentium** ili **for boja:=crna to bela do** Pazut **while** – **do**, **for** – **do** petlja može i da se ne izvrši (ako uslov u startu nije ispunjen) što smo ukoristili na slici 5 5 – program za računanja faktorijela počinje od i:=2 S obzirom da u našem primeru petlja ima samo jednu naredbu, **begin** – **end** kombinacija ovje bila potrebna.

```
y:=1;
for i:=2 to n do
    y:=y*i;
```

slika 5.5.

Naredba CASE

U programiranju se često javlja potreba da se, se svaku od nekolike dopuštenih vrednosti neke promenljive, i svrši neka od predviđenih akcije Najdlaščiji primer bio bi izbor jedne od stavki nekog menija na primer-

```
program meni (input,output);
var izbor: integer;
    dalje: boolean;
begin
  dalje :=true,
  while dalje do
```

```
begin
    writeln;
    writeln('1  Unošenje podata
    writeln( 2  Ispisivanje p
    writeln( 3  Brisanje podata
    writeln( 7  Testiranje mod
    writeln( 0  Kraj reda ),
    writeln;
    write  (    Već izbor? ');
    readln (izbor)
    if izbor=1 then writeln('Treba u
podatka )
       else if izbor=2 then writeln(
da ispisuju |
       else if izbor=3 then wri
podataka )
       else if izbor=7 then
writeln| Testiranje modela=)
       else if izbor=0 then dalje :=false
       else writeln ( Nelspravan ulaz ).
  end;
  writeln ('Kraj rada');
end.
```

slika 5.6.

Ispitivanje je, kao što se vidi, obavljena naredbom **if ... else if ... else** naredbi, što je korektno i racionalno ali se praktična programera na kad preporučljivo rešenje bolje je koristiti naredbu case.

```
    case izbor of
        1: writeln( Treba uneti podatke
        2: writeln( Podaci se ispisuju'
        3  writeln( Brisenja dele
podataka ),
        7  writeln( Testiranje modele')
        0  dalje :=false,
    else writeln ( Nelspravan ulaz'),
    end { case }
  end,
  writeln ( Kraj reda ),
end
```

slika 5.7.

Ukratko rečeno, svaka od unetih vrednosti rezultuje izvršavanjem jedna ili (uz **begin – end**) nekoliko naredbi. Primetimo, međutim da nam zastupljene sve vrednosti, predviđali smo da ulazna promenljiva može da bude 1, 2, 3, 7 ili 9 dok će bilo koja druga vrednost izazvati izvršenje **else** grane (**else** u case strukturi je ekvivalent *Turbo Pascal*-a, standardni paskal ne predviđa ovu mogućnost, dak su se autori velike drugih implementacija opredelili za reč **otherwise**). Sama **case** struktura ne počinje se **begin**, ali se obavezno završava se **end**.

Naredbe GOTO i HALT

Kažu da je paskal nastao kada su uvidele kolike je na redba **goto** štetna sva je konstrukcija okrivljena za nepregledne programe pisane u takozvanom „špageti stilu" tj programe koja je teške razumeti, testirati i modifikovati. Paskal, ipak, zadrža gotо naredbu, ali će nekoliki razlozi ubeđenika svakako predložiti da je zaboravite.

```
program skokovi(input,output);
var a  integer;
label početak, kraj, dalje, trš;
Pegin
početak:
    readln(a),
     if a=10 then goto kraj;
     if a=3  then goto trš
     if a=16 then goto dapet;
     writeln ( Nepredviđena vrednost'
     goto početak.
trš  writeln ('Trš'),
     goto dalje;
desat
     writeln ('Deset');
dalje
     goto početak;
kraj
     writeln ('Uspešan završetak');
end.
```

slika 6.6.

Program sa slike 5.5 ilustruje primenu naredbe **goto** zapra ane smo definisali nekoliko labela, a onda ih referen cirali u **goto** naredbama Ovakvi programi su veoma ru žni, ali je **goto** ponekad korisna naredba recimo kada treba „nekoliko" iz čitavog niza „ugnježdenih" petlji I pak, varujema da će sva ova naredba razmerno retko koristiti- pe ju nećemo posvetiti posebnu pažnju

Naredba (ugrađenje PI bile redi procedura) **halt** pred stavlje ekvivalent (instrukciju STOP' vedina drugih programskih jezika nalazak na **halt** prekida izvršavanje programa i predaja Pcatrolu operativnog sistemu Iako ova naredba nije predviđena standardom (jedini izuzetak" način da se završi program je nailazak na finalni **end.**) pojavljuje se praktične u svim implementacijama, pe i u *Turbo Pascal* u Tu je i jedan dodatak ako program završava sa **halt(1)**..DOS-u će biti vraćen tav **exit code** 1..na običan način bata mogli da napišete i halt(6) (vraća DOS-u kod 5 Ovi izlazni kodovi dozvolje mogu da se testiraju DOS naredbama relik ne IF ERROR LEVEL 1 – ako nste pruučavali DOS-ov komandni jezik, zapamtite samo da program treba završiti se **halt** ili **halt(0)** ako je sve prošlo u redu a sa **halt(1)** ako je sti krivena neka greška zbog koje program ne može da pravilno obavi svoj zadatak.

6. ARITMETIČKI I LOGIČKI IZRAZI

Najvažniji elementi izvršnog dela programa su aritmetički ( )logički izrazi Počinjmo od najjednostavnijeg izraza a:=b kojim se vrednost b prenosi u a. Simbol := je o vezan do Pi se dodeljivanja razlikovale od poređenja – a:=3 je naredba koja dodeljuje vrednost 3 promenljivoj a, dok je a=3 deo logičkog izraza kojim se ispituje da li je sadržaj promenljive a bal Prog 3 Promenjena vrednost se ne ograničava samo na skalarne promenljive, ali se uvek zahteva da izraz sa desne strane znaka jednakosti bude istog tipa kao i promenljiva sa leve strane – pogledajte primer 6.1

```
program dodela;
type niz = array [1, 4] af integer;
var  a  niz,
     b  niz,
     c  array (1  4] af integer,
begin
   {
     Formiranje niza b

   }
   a:=b,
   b:=b;  { ) U svoj linlji je greškal )
end.
```

dejan ristanović • moja škola paskala 13

Definisali smo nizove A i B istog tipa a zatim, umesto da pišemo **for – do** petlju, prenosi elemente niza B u elemente niza A, primenom jistog prostog naredba a:=b Da bi a isto stično bilo moguće A i B moraju da budu definisani na potpuno isti način što znači da će paskal kompajler odbiti da prevede naredbu a:=a, smatrajući ali istinito. Zatvor za „strogo ste" definicije će vam uzbiljno zamstati: samo u jednom slučaju su zmste da dodjeljujete vrednost racunajuig izraza celobrojnoj promenljivoj, pošto paskal „ne zna" automatski da celobr decimale Kontrola, dakle, funkciju trans: ar i:=trunc(a)

Za razliku od fortrana i bejzika, paskal razlikuje operacije sa celim racionalnim i Bulovim brojevima. se celim brojevima „rade" operatori +, * ceod i div za razmatra su zadatke: + , * i / a se Bulove **and, or, not**, a na *Turbo Pascalu* zar Pošto svi dobro znamo da plus označava sabiranje minus oduzimanje zvezdica množenje a kosa crta dijenje ostaje da upoznamo operatora **mod, div, and, or, xor** i **not** a zatim i specifične dodatke *Turbo Pascal*-a

Div označava celobrojno dijenje ako je i jednaka sedam a j pet, i div j daje 7/5=1 Slično tome **mod** daje o stalak pri dijenju što znači da je i **mod** j jednaka 2

Već pogadate da **and** označava konjunkciju (logičko i), or disjunkciju (logičko ili) **xor** ekskluzivnu disjunkciju (oeki je zovu ili) a **not** negaciju Upotreba svih operatora nije ograničena samo na IF naredbe pogledajmo primer 6 2

```
program zoek (input,output);
var a  char,
    i  0  255,
    velika_sl
    sala_sleva,
    cifra,
    kontrolni_ka
    spec_znak,
    greska  boolean,

Pegin
repeat
   read(n(c),
   i :=ord (c),
   velika_sleva = (i>=65) and (i<=90),
   sala_sleva  (i> 97) and (i<=122),
   greska := i>=127,
   cifra  (i>=48) and (i<=57),
   kontrolna_karakter = i<32,
   spec_znak   not (velika_sleva ar
mala_sleva ar greske
                        ar cifra ar
kontrolni_karaktar),
   if Mala_sleva then writeir ( Sama veliko
slava! ),
   if   greske or kontrolni_karaktar
   then writtaln ( Pokušaj ponavat! )
until false;
end.
```

slika 6.2.

Program 6 2 ispituje da li a sleva cifra ili specijalni znak. Najpre je formirana logička promenljiva **velika_sleva** koja ima vrednost true same ako je c između 65 i 90–zatim promenljiva **mala_slovo** pe promenljiva **cifra** pe promenljiva **kontrolni_karakter** (kodovi 0-31) (nazval promenljiva **greška** koje označave kod većl od 127 smatra se da je c specijalni znak ako nija ni sleva ni cifra, ni kontrolan znak ni greška.

Pošto sama formirali promenljive **velika_slevo, male_slevo, cifra, spec_znak, kontrolni_karakter** i **greška**, možeme da pišemo nešto poput **if velika_sleva ar scala_slova then ...** ili možde **if kontrolni_karakter or greška then ...** – zar nije mnogo prirodnije nego gomila kodova koje decinje ni samu ne razumemo bez zavirivanja u tabelu?

Mora se reći da je *Borland-ov* kompajler odstupio od „čista" koncepcije paskala utetiko što je pomešao celobroj na i Bulove vrednosti – operatori **and, or, not** i **xor** mogu da se koriste kao logički, i u tom slučaju operatu ea vrednostima **true** i **false** ili kao celobrojni, i u tom slučaju vrše logičke operacije bit po bit ako je vrednost promenljive a=210 a b=111 (ako su promenljive, pri mera radi, deklarisane kao byte), e.=a **and** b će promenljivoj a dodeliti vrednost 43 (pretvorite sva dva broja u binarne, je primenite operaciju i na svaki od bi tova ponaosob) Uvedeni su i dodatni celobrojni operatori **shl** i **shr** koji obezbeđuju aritmetičko šiftovanje nalevo ili nadesno se naredemi Prog mesta, na primer a :=b **shr** 4. Pomenućemo još i unarni operator @ kojim definišemo pointer na neku promenljivu ili proceduru (spr **vrednpoint.=@vrednost**) upotrebi svog operatora vratićemo se pri kraju ovog uneteka.

Što se racionalnih Prejeva tiče, na raspolaganju nam je sabiranje oduzimanje, množenje i dijenje, ali, paradoksalno na i stepenovanje (treba znati da je a^ = exp(b*ln(a))), naravno sama se poništava a. Tu su, naravno, i elementarne funkcije koje ćete na sledi 5.3 prepraši kratkim opisima

abs	Apsolutna vrednost
arctan	Arktangens
cos	Kosinus
exp	e^x
frac	Racionalan deo
int	Ceo deo (rezultat je racionalan
ln	Prirodni logaritam
pi	Vrednost broja pi.
random	Slučajan racionalan broj između 0 i 1
random (n)	Ceo slučajan broj iz intervala (0 n)
round	Zaokruživanje (rezultat je longint)
sin	Sinus
sqr	Kvadriranje
sqrt	Kvadratni koren
trunc	Odsecanje decimala (rezultat je longint)

slika 6.3.

U aritmetičkim izrazima javljaće se brojna konstante koje se zadaju prema standardima paskala. Racionalni Prejevi se razlikuju od celih po tački i/ili eksponentu pe čema, na primer, broj 1 2*10^15 pisati kao 1 2E12 ili. ekvivalentno 1 2e12 *Turbo Pascal* dopušta i zadavanje celobrojnih konstanti u Šekandekadnom Prejsom notaci ma, sa šta se koristi prefiks $ na primer $A1 ili $FFFFFFFF

Relacioni operatori =, <>, <, >, <= i >= dakuju na način uobičajen i u većini programskih jezika. Prioritet su međutim pomalo čudna rešeni što najbolja ilustruje naredba **if a=b and a=d then...** – pošto su logički operatori **and** i **or** po prioritetu isjednačeni sa množenjem i sabiranjem, računar će pokušati da izračuna b **and** c, pe da zatim ispita da li je rezultat jednak a. *Turbo Pascal* nema ništa protiv toga, ali će se pokazati kte se tako dobijeni rezultat poređenja ponovo pored: sa celim brojem što je sredne okolnost – da je naredba pragiaena sa očekivanim ispravnu program Pi radio zaavim pogrešna, za razlika koja Pi rada koji očekuje razumeti Ikla kako bilo logički izrazi u paskal programima su uvek prepuni zagreda- šta znači da ćemo ponovuti naredbu pisati kao **if (a=b) and (a=d) then...** Ukoliko ste u dilemi ako prioritete operacija, uvek je bolje da stavite zagrada – od njih šteta ne može da bude!

Nenumerički izrazi

Kako se u paskalu redi sa promenljivima koje nisu ni celobrojne ni racionalne ni logičke? Krajnje je vreme da upoznamo funkcije **ord, pred, succ** i **char**

Pretpostavimo da smo kao i nekoliko puta ranije deklarisali tip boje sa, **type boje = (crna, bela, crvena, zelena, žuta)** „brojevi" **crna, bela, crvena, zelena** i **žuta** su primarne, a ovo naravno sarkazam u rastućoj poredak, pa će **ord (crna)** deti vrednost 0 (prvi odnosno nulti „broj") dok će **ord (žuta)** biti četiri. Možemo o sam tagu, da koristimo funkcije **pred** i **succ** koje pronalaze prethodni odnosno sledeći „broj" očito je **succ (zelena) = žuta** i **pred (zelena) = crvena.**

Kada se redi o alfanumeričkim **(char)** podacima, **ord** daje ASCII kod karaktara o **char** pretvara ASCII kod u karakter. Nekoliko ilustracija ovih funkcije nudi slika 6.4

```
program karakteri(input,output);
{  ispisivanje ASCII seta  }
var znak  char;
    i     0  255;
begin
   writeln ('ASCII set  );
   for i :=32 to 126 do
      begin
         znak :=char(i);
         write (z
      end
   writeln;
   repeat
      write ('Unesi z
      readln (znak);
      i := ord (znak);
      writeln ( ASCII
                znak,'
   until znak= @
end
```

slika 6 4.

Rad sa stringovima

Sa mnogo pravu se može tvrditi da je rad sa alfanumeričnim Ahilova pete standardnog paskala – jezik u kojem svaka petljanje sa interaktivnu unosenim tekstom zahteva „kolotumbarski" kod prepun raznih programerskih trikova teško može da se proglasi pogodnim za praktično programiranje! Zato su razni autori raznih pocket kompajlera predvideli razna (i na žalost, međusobno slabo kompatibilna) ekstenzija **Borland**-ove ekstenzije predstavljaju dobro zaokruženu celinu, sarečete pogodnu za pomavanje bajaka, koji će se osetiti „kao kod kuće"

Alfanumerici se sećamo se, definišu sa **string[n]**, gde je **n** maksimalna dužina stringa koja se ona prilazita 255 znakova ako se a nestavi podrazumeva se 255 (*Borland Pascal i Turbo Pascal for Windows* nudi i takavsane stringove terminisane nulom koji su kompatibilni po ugledu na C i mogu sadržati znatno duže tekstove, ovde se nećemo kaviti tim dodatkom) Učitavanje i štampanje alfanumeričkih promenljivih se ne razlikuje od učitavanja i štampanja promenljivih bilo kog skalarnog tipa – možemo da napišemo **readln(tekst)** pa će promenljivoj **tekst** biti dodeljen skup znakova unesenih preko tastature U sam string spada i dužina, šta znači da ne moramo brinuti o dužini alfanumerika niti uvodi-

ti posebne promenljive – računar uvek zna koji je deo maksimalnog deklarisanog prostora zauzeti

Stringovi se mogu nadovezivati (stručni izraz je **konkatenacija**) i to primenom običnog plusa – program 6.5 će nadovezati reč "Pusel" na reč "Turbo" i tako kreirati string Turbo Pascal

```
program konkat (input, output);
var s, b, c  string;
begin
  s:= 'Turbo  ;
  b := 'Pascal ;
  writeln (s+b)
end.
```

slika 6,5,

Dopuštene je i korišćenje relacionih operatora =, <>, <, >, <= i >= pomoću kojih možemo da ustanovimo da li su stringovi jednaki ili je neki od njih „veći" od drugog; pri poređenju se, kao i u bejziku uzima u obzir isključivo ASCII tabela Ako znaci da je tekst 'Pascal' „manji" od teksta pascal i to isključivo zbog razlike i između velikih i malih slova

Suprotnost konkatenaciji je izdvajanje nekog dela stringa Ako je taj deo neka slova stringu pristupamo kao nekom nizu, opr **tekst[6]** je peto slovo stringa **tekst.** Za složenije izdvajanja koristi se funkcija **copy** ako je **tekst ='Turbo Pascal' copy(tekst,7,6)** će izdvojiti šest znakova počevši od sedmo pozicije dakle reč Pascal. Preporučujem je se ovaj način izdvajati i jedino ovde, jer **tekst[6]** se vodi računa o trenutnoj dužini stringa – ako je stvarna dužina 4, biće vraćen neki slučajno i besmislen sadržaj dok će **copy(tekst,4,1)** vratiti prazan string Ako se poslednji argument izostavi, copy vraća deo od zadate pozicije pa do kraja stringa.

Pre nego što nabrojite neki podstring morate ga pronaći Za to služi funkcija **pos** – **pos('sal', tekst)** koja će vratiti 10, jer se u stringu **tekst** „reč" sal nalazi počevši od desete pozicije Ako tekste nema **pos** vraća nulu.

Funkcija **length** vraća dužinu stringa u datom primeru bi **length(tekst)** vratilo broj 12 šeti rezultat dakle bi se se **ord(tekst[6]);** ali funkcija **length** se smatra „lepšim" programiranjem

Često su potrebne konverzije brojeva u stringove i obratno Postoje **str (broj, string)** konvertujemo su numerički podatak bilo kojeg tipa u odgovarajući string Konverzijom može da se upravlje na način koji ćemo upoznati kada budemo opisivali **write** naredbu ukratko, sa **broj** n, dine nalažemo da se broj treći veri u string dužine n, za nabrojanje dan mesta se dovoljno

Obrnuti posao obavlje procedura **val** – sa val (**string, broj i**) konvertujemo **string** u **broj** i'oite te procedura ne mora uvek uspeti uvesten je i flag i tipa **integer**ič označava da je konverzija uspela dok će ako konverzija se ospe, i vratiti poziciju prvog znaka koji je pravio problema Vredi pomenuti da se funkcija val ne može koe u našem bazicuma koristiti za izračunavanje vrednosti kompletnog izraza koji je zadat kao string; mogu se konvertovati jedne konstante sadate u okviru an omtaksom *Turbo Pascal* a.

Pomenume još procedure **insert** i **delete**, koje omogućavaju ubacivanje jednog stringa u drugi odnosno brisanje nekoliko znakova „iz sredine" stringa U praksi se razmerno retko koriste Ješ kao i funkcija **upcase** koja pretvara (jedno!) zadate male slove u velike

7. PROCEDURE I FUNKCIJE

Procedura i funkcije (ili, popštenu govoreći, potprogrami) su osnove moderne programiranja – problem koji rešavate treba najpre da podelite na nekoliko faza (upr unose podataka, testiranje ispravnosti izračunavanja rešenje i ispis rezultata) što se svodi na pozv nekoliko procedura koje te faze traturaju Zatim svaku od tih faza (procedura) ćelite ne još sitnije delove, pozivajući procedure niže nivoa, i tako dalje, dok os dodete do zaprsta jednostavnih operacija koje je lako implementirati Ovaka pisan program će proraditi mnogo brže i njegova dotnje izmene biće mnogo lakše – nema nižeg gurog od glavnog programe koji se prostire na desetak strana! Zato ćemo u ovom umetku posvetiti puno pažnje potprogramima. Krasnno od najjednostavnjeg – prenosa argumenata.

Procedure i argumenti

```
program tpot_proc(input,output);
const maxn = 100;
type  niz = array [1  maxn] of integer;
var   x,y  nin,
      pec, krej),i,j  integer;

procedure ispis (p  niz, prvi,
integer);
begin
   i :=prvi;
   while i<=zadnji
      begin
         write
         i :=i
      end
end;

begin
   for j :=1 to
      begin
         x[j] :=j;
         y[j] = j
      end;
   write  ('Od koeg se elemenata ispis
   readln (pec),
   write  ( Pa koeg se elemente
   readln (krej),
   writeln ( Elementi
   ispis (x,pec,krej);
   writeln ('Elementi
   ispis (y,5,10)
end
```

slika 7 1,

Slika 7 1 prikazuje zasvem jednostavnu proceduru koja ispisuje „komadić" niza A. Procedura je pozvana prostim navođenjem imena, dok je za njenu dešinsanje korišćena složenija naredba **procedure** Šta, naćutimo, označava tekst u zagradama?

Procedura se sluke 7 1 se zove **ispis** i ima tri takozvane fiktivna argumenta **a, prvi** i **zadnji** a je niz a **prvi** i **zadnji** celobrojne promenljive Zato ove promenljive zazivamo **fiktivnim** argumentima? Spustimo pogled sekakve redove niže i pročitajmo naredbu kojom je procedura pozvana **ispis(x, pec, krej)** Paskal kompajler će tajare da pronađe proceduru i sada da vporedi argumente

a	prvi	zadnji	fiktivni argumenti
x	pec	krej	stvarni argumenti

Argumente vidimo, moraju da budu saglasni po vrsti i redoslednu: niz x odgovara niz a, celobrojnoj promenljivoj pec celobrojna promenljiva **prvi** i celobrojnoj promenljivoj **kraj** celobrojne promenljiva **zadnji** Vrednosti stvarnih argumenata se dodeljuju fiktivnim, e onda sa izvršave procedura. Zašto čtava ova komplikacije? Zato što zasno par naredbi dovolje pozvovo mo proceduru **ispis,** ali sa drugim stvarnim argumentima **ispis(y, 5, 10)** Slika 7 3 prikazuje kako su svoga puta vrednosti stvarnih argumenata dodeljene fiktivnim

a	prvi	zadnji	fiktivni argumenti
y	5	10	stvarni argumenti

slika 7.3.

De li vam je sada jasno zašto su argumenti neophodni? Zahvaljujući njima smo ispi ili u mogućnosti da istu proceduru koristimo za ispisivanje elemenata nekoliko različitih nizova Procedura **ispis** možemo, dakle da koristimo na mnogo mesta u razmim programima za razmišljajući više o tome kako ona radi – što je ispis složenijih procedura, to će nam se više ispisiti da je jednom zauvek detaljno testiramo i dalje koristimo kao „crnu kutiju", nezajući sama ulazno parametre argumenta U tome je i smisao strukturiranog programiranja.

Pošto smo razumeli mnamo, obratimo e ale pažnje na sintaksu. Fiktivni argumenti se navode u zagradi, razdvojeni tačkama i zarezam Svaki argument mora biti deklarisan na način koji smo već upoznali – dvotačka i oznaka tipa Samo se po sebi razume da mnemo ograničeni samo na skalarne tipove. u programu 7 1 smo prenosili jedan niz e ne bi bilo problema ni da prenesemo neku promenljivu tipa **processor** ili **boje** Ukoliko promenimo više promenljivih istog tipa, možemo da ih razdvojimo zarezima – baš kao kod **var** sekcije programa.

```
procedure ispis (a  niz; prvi,zadnji
integer),
begin
   i :=prvi,
   while i<=zadnji do
      begin
         writeln(a[i] ,
         i :=i+1
      end,
   prvi,=0;
end;

Od koeg se elemenata
Pa koeg se elemente
Elementi nize X
                7
                8
                9
               10
               11
               12
Elementi nize Y;
                9
                9
                7
                6
                5
               10
pose   7
```

slika 7 4,

Hajde da malo premenimo proceduru **ispis** i dodamo joj

narodbu **prvi=0** koja će se, kao na slici 7.4, naći unutar **end**-a. Dopunili smo i glavni program naredbom za ispisivanje promenljive **pos**. Izračunavanje računanje ispisa će se **pos** jednako 7! Izračunavanje je nastalo zato što

Sama trenutak! Da li vas iznenađuje ispis **pos**=7? Trebalo bi da vas iznenadi! Jako je promenljivoj **pos** dodeljena vrednost 7 pozivom procedure **ispis** je vrednost **pos** „presuta“ u fiktivnu promenljivu **prvi**, a dodatno je vrednost promenljive **prvi** promenjena na nulu, trebalo bi da se i **pos** promeni! Ipak, sadržaj **pos** se nije promenio paskal je prepisao sadržaje stvarnih argumenata u fiktivne ali je „zaboravio“ da, izlaskom iz procedure, pregleda vrednosti fiktivnih promenljivih u stvarne. Zaboravim mnogo, u stvari nam nikako naredni ovo prepuštanje! Pogledajmo sliku 7.6

```
program proba(output);
var a,b,c,d,x,y: integer;

procedure nista (var a,b,c,d  integer
                        x,y      integer);
begin
   a :=0; b :=1; c :=2; d:=3;
   x :=4; y :=5
end;

begin
   a :=7;  b :=8;  c :=11; d :=13;
   x :=15; y :=17;
   writeln ('Pre ulaska u procedu  .
   while  ('a= ,a :3,  b=',b :3,  c= ,c :3,
   writeln ( ,d= ,d :3,  x= ,x :3,',y= ,y :3);
   nista (a,b,c,d,x,y);
   writeln ('Po izlasku iz procedure.
   while  ( a=',a :3, ,b= ,b :3, ,c= ,c :3,
   writeln ( ,d= ,d :3,',x=',x :3,',y=',y :3)
end
```

```
Pre ulaska u proceduru:  a=  7,b=
13,d= 13,x= 15,y= 17
Po izlasku iz procedure: a=  0,b=  1,
2,d=  3,x= 15,y= 17
```

slika 7.5.

Procedura **nista** se sliči 7.6, kao što joj i ime kaže, ne radi ništa pametno: samo menja vrednosti fiktivnih argumenata. Glavni program, sa druge strane, dodeljuje početne vrednosti i poziva procedure, ispisuje rezultate: kako su vrednosti promenjene a neke nisu. Kako to? Odgovor pruža pogled na deklaraciju **procedure** prepisivanja **a, b, ... d** su ispisane kao var-a promenljiva **x** i **y** nisu. Potprogram je mogao da promeni vrednosti prvih ali ne i drugih. Ukoliko, dakle, procedura ima neke **lažne** veličine (tj. promenljive koje vraćaju rezultate iz procedura u glavni program) odgovarajuće fiktivne argumente deklarišemo se var: ukoliko su promenljive uključene slažene (tj. služe za prenos podataka iz glavnog programa u proceduru) reč var neutralizuje. Korišćenje ulazno-izlaznih argumenata (tj. promenljivih koje će dopustiti podatke i preuzimati rezultate) nije baš u duhu strukturiranog programiranja, ali ponekad je teško se ovo ne koristi. Ako se želi „zaboraviti“ na ovo pravilo, treba se držati pravila: sve promenljive koje menjaju vrednosti deklarišete se var

Nemoj potreba da nabrajanjem prenošenja parametara ostane tajna. Iako često ga koristiti ako se rasumete. Pri svakom pozivu neke procedure svi argumenti koji nisu deklarisani kao var se prepisuju na stek. Kada se izvršavanje potprograma okonča, vrednosti se steka se vraćaju u nemu promenljivih, čime se anuliraju sva izmene koje je procedura prosledila nad argumentima prenošenim po **vrednosti** (to je visokestrukan izraz) Što se var promenljivih tiče, njihove se vrednosti ne prepisuju, već se procedure dostavljaju adrese stvarnih promenljivih – svaka promena jekog argumenta proustog po **imenu** (epet visokestručan izraz) prosleđivanje i kraj izvršenja potprograma!

Čini nam se da je došlo vreme da razumete jednu malu ali veoma čuvenu osnafaku nazvanu Potprogrami se zovu (**call**) i argumenti prenose po imenu (**by name**) ili po vrednosti: (by value) Kažu da je neki novinar pitao Niklausa Virta (**Niklaus Wirth**) tvorca paskala, kako se na zapravo zove (mislećI na upotrebnu prasimena) Virt je odgovorio „Sir, you can CALL me BY NAME – Wirth or BY VALUE – Werth (worth = vredan)“

Šalu na stranu kojje način prenošenja argumenata bolji? Kao i obično precizan odgovor ne postoji: da je jedan od načina uvek bolji drugi ne bi al postojao! Treba, zapak, da primetimo da prenošenja **po vrednosti** zahteva rezervaciju dodatnog memorijskog prostora, što znači da šeta, ako radite sa nekom velikom matricom dobro vam niti ako je deklarišete kao var (bez obzira što procedura ne menja njeo sadržaj

Lokalne promenljive

Najteže je proći – kako reč ne zvučali kako se potprogrami pozivaju i kako se argumenti prenose **po imenu** i **po vrednosti**, razumeste što su lokalne promenljive i ključne stvari. Ako niste razumeli – probajte da pročitate tekst još nekoliko puta, a onda nastavite program se sli ka 7.5 i pažnjom proučite rezultate

```
program proba(output);
var a,b,c  integer;

procedure promena (var x  integer
                      y: integer);
begin
   x :=5; y :=5;
   c :=303
end;

begin
   a :=1;  b:=2; c :=3;
   writeln ('Pre ulaska u proceduru  ');
   writeln ('a=',a :3,', b=',b :3, ', c=',c :3);
   promena (a,b);
   writeln ('Po izlasku iz procedure:');
   writeln ('a=',a :3,', b=',b:3,', c=',c :3)
end
```

```
Pre ulaska u proceduru.  a=  1, b=  2, c=  3
Po izlasku iz procedure: a=  5, b=  5, c=303
```

slika 7.6.

Na slici 7.5 vidimo kratak paskal program i rezultate njegovog izvršavanja, koje ćemo pokušati da proksmentarišeme Definisali smo celobrojna promenljiva **a, b i c**, a onda im dodelili vrednosti 1, 2 i 3 raspaktivne Sledi poziv procedure **promena**, čiji su stvarni argumenti **a i b** a fiktivni **x** i **y**. U zkviru procedure, fiktivnim argumentima **x** i **y** dodeljujemo vrednost 5, ali je po povratku u glavni program samo a promenjeno – b je, prenošenjem prenoto **po vrednosti**. Da sada nismo pomenuli ništa nevo Pogledajmo međutim promenljivu c – ona uopšte nije prenela u proceduru **promena**, a opet je procedura **promena** uzela slobodu da promeni vrednost c neva je vrednost, štaviše, „predivela“ i kraj izvršavanja procedure!

Procedura, dakle može da analizira i menja vrednost promenljivih koje nisu prosecna kao argumenti – ta kve promenljive zasivaju **globalnim** Tu će ju takav rad moguć na znači i da je preporučljiv – ukoliko ćete promenljiva kaju procedura menja prosek kao argument.

```
program prosti_brojevi (input,output);
const brmjava = 10;

var  x,y,i  integer;
     uspeh  boolean;

procedure prost (x  integer;
boolean);
{  Program se karakkino radili tak kada
Nema
var i  integer,  ili, jme kmlje;  var i,
integer;
}
begin
   uspeh :=true;
   y :=trunc(sqrt(x*1 0));
   i:=2;
   While (i<=y) and uspeh do
   begin
      if x mod i = 0 then uspeh :=false;
      i :=i+1
   end
end;

begin
   i :=1;
   while i<=brmjava do
   begin
      writ  ('Unesi broj: ');
      readln (x);
      prost (X,uspeh);
      if uspeh then wri
prost( )
                  nism writeln (
prost( );
      i:=i+1
end
end.
```

slika 7.7.

Globalne promenljive mogu da budu prilično opasne: zašto program se slika 7.7 ne radi kako bi trebalo? Kako bi program pre svega, uopšte trebala da radi? Sa tastature se, jedan po jedan, učitava deset brojeva, pa se, uz korišćenje kratkog i loše algoritma, proverava da li su neki od ntkucanih brojeva među prosti Primetimo da se poduci učitavaju u petlji for i =1 to 10 da, a da se u proceduri **prost** takođe koristi promenljiva i čija će vrednost, dakle, biti uništena. Zpač i: te da moramo strogo paziti na imena promenljivih koje upotrebljavamo u glavnom programu i potprogramima? Tamna polja – paskal nam omogućava da u programu i potprogramu koristimo ista imena promenljivih na pažnji se zatuvd Da bismo se uverili u to dopunimo program se slika 7.7 jednom deklaracijom **var i:integer;** koju ćemo ubaciti neposredno iza procedure **prost**. Rezultat? Program se korektno prvodi i **korektno radi**. Što je za početnika prilično čudno – kako to da se konapajler na „bunu“ Što dva puta definišemo promenljivu i?

Procedura je kao što smo već istaljive napomenuli potpuni paskal program koji može da ima svoje konstante, svoje promenljive pa čak i svoje procedure i funkcije U tom smislu on iza **procedurea ime(argumenti)** može dodati sekcija konstanti sekcija tipova i sekcija promenljivih. svako deklarisano konstante i promenljive vaše samo u proceduri koja ih je definisala i eventualne, u

proceduruma koje je ta procedura dalja pozvala – to se takozvane **lokalne promenljive**. Lokalne promenljive mogu da imaju jednakvana imena ti da nase imena nekih promenljivih glavnog programa ili drugih potprograma – može doći ni da kakve zabune Lokalna promenljiva vam omogućavaju da procedura piše i testirate potpuno ne zasavisna od glavnog programa.

```
program testiranje (input,output);
{  Pegedim rezultate!  }
var y,z,i,j  integer;

procedure test1 (var a integer);
var i,j  integer;
begin
   i :=4, j =7,
   z =3, a =5,
end;

procedure imet (m  integer),
var i,k,z  integer,
begin
   test1 (i),
   writeln ( i= ', i:3),
   a :=4,
   i :=0  j :=905
   test1 (k),
   writeln ('k= ', k:3)
   writeln ( a= ', a :5),
end,

begin
   x :=55, i :=55, j :=90,
   y :=303,
   imet(y),
   writeln ('z= ',z:5)
   writeln ('y= ',y:5)
   writeln ('i= ',i:5)
   writeln (
end.
```

slika 7.8.

Slika 7.8 treba da vam omogući da proverite stečena znanja – pokušajte da predvidite rezultate izvršavanja programa a zatim ga startujte i proverite svoje zaključke Ukažko se pojave razlike, pokušajte da pronađete greške u nepotvrnom razmišljanju take se najbolje učiti

```
  writ[illegible]
end;

begin
  x =88, i =88, j =88,
  y =383,
  test(y),
  writeln ( x=  ,x ),
  writeln ('y=  ,y ),
  writeln ('i=  ,i ),
  writeln ( j= ',j )
end.
```

slika 7.9

Pošto smo „savladali" sliku 7.9, prelazimo na primer 7.9, koji je nešto (ali samo nešto) složeniji od primera 7.8 u proceduri **test1** – u primeru 7.9 je naročito **proceduru test** iz, a u primeru 7.8 upred definicije potprograma **test1**. Pomenuo reč je, međutim, bitno promenio značenje programa, o čemu ćemo govoriti nešto kasnije. Pogledajmo najpre rezultate na slici 7.9 je **test1** nazvana procedura sa lokalnom promenljivima, tj. procedura koja se može da manipuliše sa lokalnim promenljivima procedure **test**. Procedura **test1** na slike 7.9 je, na drugu stranu **lokalna procedura za test** što znači da može da operiše sa promenljivom Z! Razlike je i u tome što glavni program na slika 7.9 može direktno da poseve proceduru **test1**, dok je na slici 7.9 to nemoguće – samo **test** može da poseve **test1**!

Da li je koje karakteri lokalna procedura (koja su eksklusivno vlasničke drugih procedura) di učinili da svaki potprogram bude nezavistan od drugih? Ukoliko procedura obavlja neku stvarno specifičnu radnju koje je samo za sebe potpuno beznačajna, projektuje je za potprogram nekog drugog potprograma. Ukoliko, na drugu stranu, procedura ima neku jasnu svrhu koju će sveotualno moći da koriste i drugi potprogrami, projektuje je za global nu!

Funkcijske naredbe

Funkcije su prilično slične procedurama – može se reći da je funkcija specijalna procedura koja ima nekoliko ulaznih i **tačno jednu** izlaznu vrednost. Navimo se, naravno najbolje razumeju na primeru.

```
program faktorijel (input,output);
const max=12;
var  n  longint,

function fakt (u longint)  longint,
var i,y  longint,
begin
   y =1,
   for i =2 to n do
      y =y*i;
   fakt.=y
end,

begin
   write  ('n=');
   readln (n),
   if u>max then writeln ['Prevelike n  i
                 else writeln (n ,'!=',fakt( i
end.
```

slika 7.10

Pogled na sliku 7.10 treba da nas podseti na program za računanje faktorijela na slike 6.3. Skoncentrišimo se na prvi novitet, **function fakt**(**u**: **longint**):**longint**; **Function** je službene reč **fakt** je ime funkcije N je argument (i to celobrojan zbog **u**: **longint**) ali šta bi mogle da označava ponavljanje reči **longint** iza zatvorene zagrade? Svaka funkcija vraća jednu vrednost, pa **moramo** da kažemo kojeg je tipa ta vrednost – u našem je slučaju faktorijel ceo broj pa smo napisali da je funkcija **fakt** celobrojna tj. da vraća intidžer

Iza imena funkcije prenašamo deklaraciju lokalnih promenljivih i y (i funkcije mogu da imaju lokalne promenljive, lokalne konstante, pa čak i lokalne funkcije i procedure veže svi navedi i sva ranije izložena upozorenja), za kraju **begin** – **end** struktura koja računa fak torijel. Primetimo, pak, da se funkcije završava na **fakt.=y**; na taj način stavljamo kompajleru na znanje da je vrednost koja treba da se vrati u glavni program a a on i ih možda **y+1**. Vrednost funkcija se dakle, na kraju obavezno dodeljuje njenom imenu tako znamo da nas nećete poslušati, navratićemo vam da **nikada** ne pokušate da „uštedite" promenljivu Y koristeći ime funkcije kao „običnu" promenljivu – takav postupak predstavlja prave pravcate traganje za nevoljama!

Zanimljiva je da standardni paskal omogućava da imena procedura i funkcija budu argumenti drugih procedura i funkcija ali da to ne *Turbo Pascal* u nije dopuštene – sva je jedna od retkih stvari u kojima je *Borland*-ov kompajler slabiji od standarda.

Rekurzija

Na slici 7.8 smo videli da samo nikakve prepreke da se neka procedura (ili funkcija) poziva neku drugu (ranije definisanu) proceduru ili funkciju i take da prilično velike dubine, koja zavisi uključivo od raspoloživa memorije. Mogućе je takođe da procedura ili funkcija pozove sa mu sebe, čime dolazimo do pojma rekurzije

Ne prethodnim smo stranama napisali bezbroj programa koji računaju faktorijele množeći sve prirodne brojeve manje od zadatog Matematički razmotrimo činjenicu da primetiti da izraz $1*2*3*\ \cdot N$ nije prava definicija faktorijela, pravilnije je napisati

N!=1, za N=0

N!=N*(N-1)!, za N>0

Pokušajmo da dokažemo da je 3! isto što i $1*2*3$ Po gornjoj definiciji je $3!=3*2!$ Dalje je $2!=2*1!$ a $1!=1*0!$ Ol je i po definiciji, pa je onda $1!=1*1=1$ tada je $2!=2*1!=2*1=2$, a $3!=3*2!=3*2=6$ Ne dokaz način bi smo mogli da pokažemo da je $5!=120$ (pokušajte!) i da tako bar povećamo u gerulju definiciju faktorijela ako ne i da matematički dokažemo njenu ekvivalentnost sa ranije izloženom formulom Šaširavimo sada novu verziju funkcije **fakt**.

```
function fakt (n:integer)) integer;
begin
   if n=0 then fakt =1
          else fakt :=n*fakt (n-1)
end;
```

slika 7.11

Na prvi pogled bi se reklo da je ovako definisana funkcija mnogo bolja od one koju smo napisali pre nego što smo znali za rekurziju: nema ima i ni jedna jedina pomoćna promenljiva (ranije smo koristili dve, y i i). Mogućе, međutim da kažeme da je funkcija sa slike 7.11 već potrošila memoriju, pošto se svaki poziv treba sačuvati adresu povratka, argumenta, itd. Uspešne su programi koji koriste rekurziju nasko kratke elegantni i jednostavni, ali troše dosta memorije i računarskog vremena, pa ih treba izbegavati kada je to mogućе (p mogućе je uvek, ca više ih manje truda)

Neki rekurzivni algoritmi zahtevaju da procedura A poziva proceduru B, a procedura B proceduru A, što se može na umnaze da uradimo paskal (za dva izuzetka do kojih još nemo došli) zahteva da ono što koristimo obavezno bude već definisano Ovaj se problem rešava kao na slici 7.12, korišćenjem služ bene reči **forward** napi šemo ime funkcije i njene argumente, a ostim, umesto i zvršnog dela, tagnemo samo **forward** Ostaje nam, naravno, obavezu da negde pre glavnog programa dopi šemo i radni deo procedure.

```
program Ame;
{  Program koji uzasa
   u pravu petljnji
}
var a,b  integer

procedure qivar x
forward,

procedure p(
begin
   q(z)
end,

procedure q;
begin
   p(b)
end;

begin
```

slika 7.12.

Završavajući ovu kratku priču o rekurzijama, na možemo da ne navedemo jedan divan primer njihove upotrebe. Molda sam je pokrat problem nazvan „Kule Hanoja" prše se da u gradu Benaresu postoji podzemni hram u kojem je indukski bog Brama pri stvaranju sveta postavio tri dijamantska štapića i na jedan od njih stavio 64 zlatna kotura različitih veličine no zamiju je stavio najveći kotur na njega manju i tako sve da najmanjeg kotura koji se nalazi na vrhu tajnstvene kupe Svećenici svešteniku dana i noću prenosi koturove sa prvog na drugi štapić slušeći se trećim kao pomoćnim i poštujući ograničenje koja je zadao Brama. prateni smeju da se premeštaju samo jedan po jedan i nikada veći preten na manu da se stavi preko manjeg Kada svećenici budu obavili ovaj na prvi pogled jednostavan posao nastupiće kraj sveta (ne plašite se jednostavan račun pokazuje da nam je do kraja sveta ostalo još har 500 milijardi godi na) Nije nimalo lako sastaviti program koji rešava ovaj problem bez rekurzija rekurzivni program je, kao što vidimo na slike 7.13 prava dečja igra!

```
program Hanoj (input,output);
const osmx=64;
type  diskova = 1  0;
      raspon  = 0  osmx,
var   n  integer,

procedure prenos (n raspon, x,y,z  diskovi)
begin
   if n<>0 then
   begin
      prenos(n-1,x,z,y);
      writeln('Sa',x:3,'
      prenos(n-1,z,
   end
end
begin
   write
```

```
   if n=0 then writeln ('Čeka] se kraja
sveta!')
          else prenos (n,1,2,3);
end.
```

slika 7.13.

Ugrađene procedure

Kao što postoje ugrađene funkcije (**sin, cos, exp**,) tako postoje i ugrađene procedure, za neda ćemo upoznati **read, readln, write** i **writeln**, dak šeme se ostalim procedurama ređe kada nam zatrebaju.

Ne treba biti veliki poznavalac kompjutera da bi se na ključile da **read** predstavlja učitavanje sa tastature, a **write** ispisivanje na ekran. One ln koje se često „prikrpi" na **read** ili **write** je skraćenica od **line** razlike i između procedura **write** i **writeln** je u tome što posle izvršavanja prve kursor ostaje u istom redu (kao da smo na bezuku napisali PRINT „ABCD";), dok druga izaziva prelazak kursora u sledeći red pa navratiću upoznavanje. Situacije kod **read** je nešto složenija, što znači da je u praksi najbolje koristiti **readln**.

Argumenti procedura **read** su promenljive čije sadržaje treba učitati sa tastature. Procedura **write** je nešto kompliko vanje. njen argument, pre svega, može da bu da string pol zgustrašena koji se direktno prenosi na ekran Izuzevanjem vrednosti promenljive se na neki način može upravljati. ax4 na primer, znači da će vrednost celobrojne promenljive a zauzimati četiri pozicije na ekranu ako je broj manji od 1000 bide dopunjen vodećim blankovima. Vrednost recimo realne promenljive e može se ispisati na mi3, što znači da će vrednost zaokružiti na dve cifre iza decimalne tačke i predznak broja) pri čemu će se ispisati tačno tri decimale

Borlandovi dodaci

take su procedure i funkcije implementirane na način kaji je veoma sličan standardima paskala, *Borland*-ovi programeri su uneli neka poboljšanja. klaunule **interrupt, external** i **inline** i naredbu **exit**

Najlakše je razumeti klaunulu **external** – ako program pisan na *Turbo Pascal*u treba da pozove neki potprogram pisan na asemblaru, taj potprogram treba na neki način uključiti u finalnu EXE datoteku obezbeđujući pri tome mehanizme za poziv i prenos parametara. Fotografiju pasakalu (skaro) sve što se koristi ćemo biti deklarisano definiciju eksterne procedure ćemo zamenuti sa.

```
{$L POPUNI.OBJ}
procedure popuni (var pos  pozatak  longint
{ili fun: longint);
external;
```

Pretpostavka je da je POPUNI.OBJ datoteka sa asembliranim programom koji preuzima vrednosti argumenata i popunjava blok memorije podacima. Ovakav prenos praktikujemo samo ako je korišćen segment na početku programa nešto duk – par asemblerskih je strukcija možeme da umetnemo as **inline**, kao u sledećem primeru (laka date pogneti da je BFA kod ia strukcija CLI koje ouemogućave prekide)

```
procedure DisableInterrupt; inline ($FA);
```

Pamemmo i klaunulu **interrupt** koja omogućave definisanje interaptskih procedura Interaptska procedura ntvak mora da počinje se

```
procedure ips
{flags,cs,ip,ax,bx,cx,dx,pi,di,ds,es
word};
interrupt;
```

Da li ste primetili neke neobične u prethodnim primerima? Trebalo bi da jeste – skrećemo vam pažnju na ken strukciju **procedure popuni (var gde, početak: longint; količina: longint);** u kojoj formalni parametar **gde** uopšte nije deklarisan. Neke slično je na standardnom paskalu nezamislivo, pa stoga ovu ekstenziju *Turbo Pascal*-a pomalo mnogo znamla koristiti pomislač se, ipak, može ukazati potreba da napišemo proceduru koja će operisati sa raznim tipovima podataka. Program 7.14 ilustruje poređenje dva strukture bez obzira na ip hov tip uz pomoć procedure **equal**

```
program poredi (input, output);

type vector = array [1..10] of
     point  = record
                x, y  integer;
              end;

var  vec1, vec2, vector:
     u  integer;
     p  point;

      read(vec1[u]);
    end;

function equal (var source, dest; size
word): Boolean;
type bytes = array (0  maxint] of byte;
var  n  integer;
begin
  n =0;
  while (n<size) and (
[bytes(source)[n]] do
        inc(n);
  equal = n = size
end

begin
  read(vec1[1], 2, 4)
end.
```

slika 7.14

8. MODULI

Moduli (ili, u terminologiji *Turbo Pascal*-a, **units**) predstavljaju značajnu ekstenziju koja u mnogome umanjuje razlike između paskala i modula 2. Iako nisu predviđeni standardima paskala, moduli su osnovni u skladu sa njegovim duhom – strukturirano programiranje zahteva da se funkcionalne jedinice izdvoje u posebne funkcije ili procedure ali se standardnom paskalu te funkcije i procedure moraju biti upisane u istu datoteku u kojoj je i glavni program. Ukoliko se dakle, ukaže potreba po ponovnu u nekoj od procedura, računar nema načina nego da pravde kompletan program sajedno sa svim upravnim procedurama i funkcijama. Zar ne bi bilo prirodnije da se svaka funkcionalno povezana grupa procedura odvoji u posebnu datoteku da se osnovne testira, prevede i kada je god to potrebno, uključi u finalni program? Takva funkcionalno povezana grupa deklaracija, procedura i funkcija se, počevši od *Turbo Pascal*-a 4.0 zove modul (**unit**)

Struktura modula

```
unit ime;
interface
  uses liste;
    { javne deklo
implementation
    { privatne d
    { procedure i
begin
  { inicijalizacioni kod
end.
```

slika 8.1

Struktura modula prikazana na slici 8.1 je u mnogo čemu slična strukturi samog programa. **Unit** ime je, da počnemo od početka, (ovoga puta neophodni) ekvivalent deklaracije **program** – računaru se stavlja da znanja da je ono što sledi modul a ne rečimo glavni program a onda se tam modulu daje ime. Sledi interfejs sekcija, preko koje modul komunicira sa nadređenim modulima – svi tipovi, konstante, promenljive, funkcije i procedure deklarisane u ovoj sekciji su „vidljive" na pozivajući modul. Iuterfejs sekcija započinje službenom reči **interfaces** i završava se službenom reči **implementation** zanimljiva je da iza ove dva reči ne treba stavljati separator ; što se nekc nađa nije u skladu sa konceptom paskala.

Prva (neobavezna) linije interfejs sekcije je **uses** – uz njenu pomoć modul može da koristi druge module navedene u listi. Ostatak je sasvim sličan početku paskal programa. sekcije tipova, promenljivih konstanti i, na kraju lista procedura i funkcija. Kažemo **lista** procedura i funkcije jer se u interfejse sekciji navode samo imena i argumenti funkcija i procedura dok se njihov redni deo „skriva" u implementacionu sekciju, kljunula **forward** ovde nije potrebna niti dopuštena.

Implementaciona sekcija predstavlja privatni deo modula – ovde se pozavija deklaracija svake procedure, prepisuje njenom radnom delom planiran u skladu po sintaksom *Turbo Pascal*-a ukoliko se za tim ukaže potreba, ovde se može koristiti kljunula **forward**, što znači da se deklaracija neke funkcije ili procedure u okviru kompletnog modula može javiti čak i tri puta. Implementacione sekcija se mora da sadrži samo radna delove procedura deklarisanih u okviru interfejs sekcije – mogu se deklarisati ovi tipovi, konstante promenljive, procedure i funkcije koje su nevidljive za pozivajući program i koje „javne" procedure koriste da bi obavile svoj posao. Implementaciona sekcija, sve u svemu, predstavlja pravi paskal program koji je dopunjen (dodate neobavezanim) glavnim programom koji počinje sa **begin** p završava sa **end.** (sa obavezna tačku.) Ovaj pseudoglovni program se automatski izvršava pre pravog glavnog programa i treba da obuhvati naredbe koje će izvršiti inicijalizaciju promenljivih koje su neophodne po normalan rad modula. Nema nikakve prepreke da u okviru ove inicijalizacije postavimo i neke privatne vred nosti deklarisane u okviru Implementacione sekcije modula – one će, dodule biti nevidljive za ostatkom program ali će one vrednosti „zadržati" kada god modul bude pozvan

```
UNIT stat;

 INTERFACE

procedure oleor;
procedure uswy   (x,y
procedure fundbr (
```



```
IMPLEMENTATION

type struct   = record
                  nn, sx, sy
                  sx2, sy2, sxy
                end;

var  statbleck  struct;

procedure cleor;
begin
  with statbleck
  begin
    nn =0;
    sx2 =0;
  end
end.

procedure newxy (x, y
begin
  With statbleck do
  begin
    nn =nn+1; sx =sx+x, sy
    sx2 =sx2+x*x, sy2 =sy2+y*y
    sxy =sxy+x*y
  end
end.

procedure findmbr (var
var teep  real;
begin
  with statbleck do
  begin
    teep =nn*sx2-sx*sx;
    m =(nn*sxy-sx*sy)/teep,
    b =(sx2*sy - sxy*sx)/teep
  end
  =(nn*sxy-sx*sy)/sqrt(teep*(nn*
      r =sqs(r)
  end
end.

begin
  cleor
end.
```

slika 8.2

Primer 8.2 predstavlja ilustraciju svega o čemu smo do sada govorili – napisali smo statistički modul koji se ugrađuje u svaki program kojem je potrebna takozvana linearna regresija, odnosno povlačenje prava linije koja u minimalnoj mogućoj meri odstupa od serije eksperimentalno određenih tačaka. Modul treba smestiti u datoteku **stat.pas**, a zatim izvršiti prolaz prevodenjem – Alt R se pretvodi izvršni kod, ali je korisno i radi otklanjanja eventualnih grešaka. Pravi resultujući kod koji će biti upisan u TPU datoteku proizvešćemo prevođenjem glavnog programa iz primenom opcije **Build** odnosno **Make** iz **Compile** menija.

Upotreba modula

Moduli su koriste sasvim jednostavno – na samom početku pozivajućeg modula (kažemo **pozivajućeg zaobiđi** s ne **glavnog programa** jer i moduli mogu da koriste druge module) treba napisati **uses lista;** gde je **lista** spisak korišćenih modula razdvojenih zarezima. U izvršnom delu modula mogu se slobodno koristiti definicije funkcije i procedure pomenute u interfejs sekciji podređenog modula bez potrebe da se ono ne bilo koja naša deklaraciju. Zanimljiva se situacija javlje jednom ako se u po hijerarhiji starijem modulu deklariše neka promenljiva čije je ime jednako promenljivoj iz modula. Prednost tada ima glavni program ali je tpak omogućen i pristup promenljivoj iz modula i to sa **ime_modula.ime_promenljive**.

```
program statistika (input, output);

uses stat,

var n, i  integer,
    x, y, p, b  r  real,

begin
   write ( Kolike će perove biti?
   readln (n),
   cleor,
   for i =1 to n do
   begin
      write  ( x[ ,i,']=
      readln (x),
      write  ('y['
      readln (y);
      newxy  (x, y);
   end
   findmbr (m, b, r),
   writeln ( Koeficijent
   writeln ( Odsečak
   writeln ( Koef  korelacij
end.
```

slika 8.3

Primer 8.3 ilustruje upotrebu modula **stat** po slike 8.2 – računar učitava sarzu parova (X, Y) a onda povlači najbolju pravu kroz njih i određuje koeficijent koralacija. Ograničenja mikroprocesora 80x86/88 učinila su da jedan segment ne može pravazići 64 kilobajta što vala i po *Turbo Pascal* Moduli, međutim predstavljaju elegantan način da se ova ograničenje prevazide – ograničenje od 64 kilobajta vazi za svaki modul posebno što znači da kod više modula uzet zajedno može i te kako prevazići 64 kilobajta limit je ukupan RAM veliko kompjutera umanjen za određeni segment koji zauzimaju DOS i residentni programi. Ova tvrdnja, na žalost, ne važi za segment podataka – sve promenljive glavnog programa i svih njegovih modula se mogu zauzeti više od 64 kilobajta. Ograničenje data segmenta bez sumnje predstavlja najveću manu *Turbo Pascal*-a.

Sistemski moduli

Prilikom instalacije *Turbo Pascal*-a u radni direktorijum su prekopirane brojne datoteke među kojima vaćne mesto pezuzime fajl **turbo.tpl** Te je modul koji sadrži veoma važne deklaracije funkcije i procedure bez kojih *Turbo Pascal* uopšte ne bi funkcionisao takozvane **izvršna biblioteka** *Turbo Pascal*-a zove se **system unit** i nalazi se u datoteci **turbo.tpl** S obzirom da su rutine iz sistemskog modula stalno neophodne, spitakzam nije predviđena akavaza da se svaki program započne sa **uses system** Datoteka **turbo.tpl** medium, sadrži i druge važne module koje ćemo ovde samo nabrojati dok ćemo opširnije opis ostavili se finalne pobližje. Svaku od modula m, ukoliko ga koristimo mora eksplicitno naveati iza usesm što je prilično zgodne – modul ćemo referucirati samo oko zaista nameravamo da ga koristimom što opređava de EXE datoteke bude pretrpana kodom koji se uopšte ne pozive!

DOS unit sadrži razne deklaracija, funkcije i procedure koje programima razvijanim na *Turbo Pascal* u omogu davaju korišćenje usluga MS DOS-a odnosno PC DOS-a,

zastavni deo ovog modula su, na primer procedure **GetTime, GetData, DiskSize,** itd. Definisani tipovi i konstante ovog modula obezbeđuju pisanje interaktivnih rutina i drugih mašinski zavisnih procedura na *Turbo Pascal*-u.

Crt unit obezbeđuje opcije za očitavanje tastature, ispis teksta na ekranu i generisanje zvuka koje su precizije od standardnih procedura **readi** i **write**. Modul se, međutim, ne bavi grafikom

Printer unit je veoma kratak i obezbeđuje komunikaciju sa štampačem. Ovaj modul koristi **Crt**

Graph unit je zapravo moćan skup grafičkih rutina koje su kompatibilne sa CGA, EGA, VGA, Hercules, AT&T 1BM8514 i PC3270 standardima. Ovaj modul koristi **Crt** i nije integralan deo datoteke **turbo.tpl** – smešten je u **graph.tpu**.

9. SLOGOVI I DATOTEKE

Iako paskal primarno nije namenjen radu sa bazama podataka, njegovim standardima su predviđene razne sa učitavanje i snimanje podataka na disk, uz minimalna mogućnost obrade. *Borland* ovi programari su uvili brojne ekstenzije koje obezbeđuju ne samo slobodan pristup raspoloživ za rad sa bazama podataka već i pisanje sistemskih programa koji operišu sa datotekama on di ku i periferijskim uređajima – ono što se paskala tiče, predstavljaju najobičnije datoteke. U ovom poglavlju u glavnome ćemo se baviti tim ekstenzijama

Počnimo od sloga – na slici 9 1 vidimo deklaracije koje bi bile ugrađena u neki paskal program koji bi se bavio kadrovskom evidencijom svaki službenik ima slog u kojem je, osim imena, prezimena, pola i radnog staža upisana broj radnih sati i ukupna plata.

```
const ime_duz  = 10,
      prez_duz = 25,
      max_zap  = 100,

type  radnik = record
                 ime:     [illegible]
                 prezime: string[illegible]
                 pol:     (muskarac, [illegible]
                 staz     0  40,
                 sati     0 .300;
                 plata    le[illegible]
               end,

var   zaposleni  array [1  max_zap  [illegible]
radnik;
      umanseni   radnik;
```

slika 9 1

Tip **radnik** je definisan kao **record** (slog) koji se sastoji od polja **ime, prezime, pol, staž, sati** i **plata** svako od svih polja je deklarisano na odgovarajući način, što znači da u svakom zapisu imamo dve stringa, jednu promenljivu specijalnog tipa, dva cela broja i jedan cea broj maksimalne dužine (**longint**) Definicija sloga se, kao i sve druge strukture u paskalu, završava se end, dok su polja zasebnjena separatorom ;

Slogovi se moraju da budu vezani se spoljnom memorijom – u primeru 9 1 uvedena je niz **zaposeleni** koji se sastoji od najviše 100 slogova **radnik** – koristili smo, dakle dinamičan da elementi matrice mogu da budu na samo skalari već i kompleksnije strukture Što se operacija se slogovima tiče možeme da preneseme sadržaj jednog sloga u drugi (npr **zaposleni[i] = umseni**) ali nam je daleko važnija mogućnost da pojedinačne prustupamo zakim poljima. Kao što za prustupanje elementima matrice koristimo srednje zagrade tako za prustup pojedinom polju koristimo tačku ako je treći zaposleni muškarac zaposlenje onda pišemo **zaposleni[3] pol := muskarac**. Stvar ne postaje manje komplikovanija ako definišemo slogove čiji su elementi drugi slogovi

```
type  mes     = (jan, feb, mar, apr, maj,
   , jul,
                 avg, sep, okt, [illegible]
      datum   = record
                  den    1. 31;
                  mesec  mes;
                  godina 1980 .[illegible]
                end,
      radnik = record
                  ime      string (ime_[illegible]
                  prezipe  string (prez[illegible]
                  pol      (muskarac  zene)
                  redjen   datue,
                  otex     0  40,
                  seti     0  300,
                  plata    iengint,
                end

var   zapesleni  arrsy (1  max_zap) of
radnik,
      oneseni    radnik,

begin
   zapeslenl[3] redjen den =12,
   zapeslenl[3] redjen mesec =jul;
   zapeslenl[3] redjen godins =[illegible]
   {    }

end,
```

slika 9.2.

U primeru 9 2 tip **radnik** je proširen dodavanjem datuma rođenja, tj. je datum novi slog koji se sastoji od dana, meseca i godine Na ustoj slici vidimo kako se unosi podatak da je treći radnik rođen 12 marta 1962 godine **zapoleni[3]** nas upozorava da se radi o trećem zaposlenom **redjen** označava da radimo sa poljem **rodjen** dok **.dan, mesec i godina** označavaju pristup odgovarajućim poljima „Polja u poljima" učite čine mnoge veoma dugačkim, što je, pre svega, nepregledno a zatim i neprijatno za kucanje. U primeru 9 3 pokorstili smo na **redbu with** na početku naglasmo with **ime_sloga** da, a zatim navodimo samo imena polja kojima prustupamo Struktura **with** se, zasvim očekivano, zaključuje sa end

```
with zapeslenl[3] rodjen do
begin
   den =12,
   mesec =jul,
   godina =1962
end;
```

slika 9.3.

Navedja je što podaci u praksi nisu uzi za svaki slog. U zmisno na primer da u našem programu za evidenciju radnika treba unositi podatke o načinu uplata – ako radnik ima tekući račun, potrebno je i njegov broj kao i ime banke kod koje je otvoren ako radi preko zadruge potrebna je šifra zadruge a ako prima platu na blagajni, nije potreban nikakav dodatni podatak Ú zavisnosti od polja **isplata, dakle_ostatak** sloga može da ima tri raši čite strukture Paskal zaključi omogućave dak i svakovu vratolomiju što dinkasuje primer 9 4

```
type  ispl    = (tekuci,zadruga,gotovina),
      rednik = record
                  ime      string (20),
                  prezime  string (30),
                  pol      (muskarac  zene),
                  otez     0  49,
                  xsti     0  300,
                  plate    lengint,
                  case isplate  ispl of
                     tekuci   (
                        brc)  stri[illegible]
                        banke
[illegible]ing(20), ),

                     zadruga  (
                        zadr_sifr  word;
                        ci_broj  lengint,

                  end,
```

slika 9 4

Tipizirane datoteke

Slogovi u memoriji mogu da budu interesantni i korisni, ali su ipak nuzu „one prave" – sloj okšne permanentno se datotekama na nekom mediju spoljne memorije tj. na disku

Datoteke u *Turbo Pascal*-u dele se na tipizirane i potpuzirane dok se tipizirane datoteke dale na tekstualne i „ostale" Pređemo bal od tih „ostalih" jer su najpodesnatvnije za razumevanje i najbliže standardima paskala. Ideja je da datoteke bude u binarnom formatu i da se sastoji od slogova istog tipa – svaki od tih slogova decu je se može učitati u memoriju, promeniti i pesove sni miti na isto ih neke druge mesto Podrazumeva se, međutim da će slogovi biti čitani i upisivani samo pomoću paskal programa – sadržaj tipizirane datoteke će, ako ga upišeme DOS komandom TYPE dalevet; poljui na besmislene, a bile kakva promena tog sadržaja pomoću nekog editora može učiniti čitavu datoteku nzupotrebljivom.

Upis podataka u tipiziranu datoteku obavlja se komandom **write** a čitanje komandom **read** – na početak bate treba samo otvoriti logičko ime datoteke, na primer **wri te(izveštaj, slug)** upisuje **sloj** u datoteku **izveštaj** Mogla bi se, zapravo, reći da se naredbe **read i write** i u onovnom obliku opšte sa datotekama **input i output**, o tim što nema potrebe da se ova **mesta**, pridružena tastaturi i ekranu, eksplicitno navode

Datoteku koju ćemo koristiti moramo prvobno najpre deklarisati i otvoriti. Deklaracija je unekoliko slična deklaraciji matrice **var ime: file of tip** Ukoliko se na primer u datotaci nalaze slogovi tipa **radnik** (slika 9.2), odgovarajuća deklaracija će biti **var kadrovi: file of radnik** Take definisane ime je u otvari **logičke ti** mo značu jedino u okviru samog programa. Treba ga još povezati se konkretnom datotekom na disku, se što se koristi komanda **assign (logičko ime, fizičko_ime)** Samo se po sebi razume da **fizičko_ime** može da sadrži za samo ime datoteke nego i ime diska odnosno direktorijuma na primer **assign (kadrovi, 'dkravnovlje akal\kadrovi.dat')** dopuštene je i korišćenje string promenljive što znači da za tastature možeme učitati ime datoteke i onda na neki način obraditi njen sadržaj

Slazno povezivanje logičkog i fizičkog imena nije dovoljno datoteku treba otvoriti Ú standardnom paskalu sve su datoteke sekvencijalne što znači da su potrebne sa me procedure **reset (ime)** i **rewrite (ime)** prva otvara datoteku za čitanje i modifikovanje o druga sa upis, pri čemu se podrazumeva da pre otvaranja datoteke se upis ranju sadržaj treba uništiti Ukoliko pokušate da otvori te nepostojeću datoteku za čitanje biće prijavljena greška koju treba na vreme uhvatiti ugrađivanjem opcije {$I } u odgovarajući segmnat koda, pasle otvaranja, koristi se funkcija **IOResult** koja, vrednošću različitom od nule, signalizira postojanje greške i pruža priliku programo da ispše odgovarajuću poruku ili preduzme neke druge akcije Pošto je rad sa datotekom završen, treba je zatvoriti komandom **close (ime)**

```
type  kadrovi = file of radnik,

var   f         kadrovi,
      sleg      radnik;
      fnase     string,
      brojac    integer,
      ukplete   lengint,

begin
   write  ('Ime datoteke  '}
   readln (fnase);
   if fnase =   then halt;
   assign (f, fnase),
{$I }
   reset (f),
   if iresult<>0 [illegible]
   begin
      writeln([illegible]
      halt(1);
   end,
   brojac:=0; ukplatis=0,
   while not eof (f) do
   begin
      read (f, sleg),
      if iresult<>0 then
      begin
         writeln( Neuspele čitanje [illegible]
datoteke ),
         close(f),
         exit,
      end,
      brojac :=brojac+1
      ukplata :=ukplete+sleg.plata;
   end,
   close (f),
   writeln ( Ukupno radnika      , broj[illegible]
   writeln ('Svega za isplatu    ,
ukplats 10),
end
```

slika 9.5

Primer 9 5 je jednostavna ilustracija sekvencijalnog ra da sa tipiziranom datotekom – čitali smo je slog po slog i sabirali sadržaj jednog od polja Svako čitanje je, pri rednom, praćeno proverom statusa (**iresult**) što znači da bi se rad programa prekinuo ako na primer, dukota se podacuna ne bi bila u dravu. Kada se stigna do kraja datotaka **eof(f)** će postati **true** i računar će i zaći iz petlje upotrebi izvršitaj

Sekvencijalno čitanje je tek početak – potrebno nam je način da pročitamo i promenimo neki konkretni slog Recimo da je radniku čiji je slog deseti po redu (slogovi se uvek broje od nule tako da deseti slog zapravo nom broj devet) plata povećana za 50% program koji bi i zvršio izmenu mogao bi da izgleda otprilike ovako (u ci-

ju nikada prestara, za navedene deklaracije kao ni deo koda koji otvara i zatvara datoteku)

```
seek (f, 0);
if ioresult<>0 then
begin
   writeln( Neuspelo pozicioniranje na
slog );
   halt(5);
end;
read (f  slog),
if ioresult<>0 then
begin
   writeln( Neuspelo
   halt(1);
end;
slog.ploca :=slog
seek (f, 0);
write (f, slog)
if ioresult<>0 then
begin
   writeln( Neuspeo upis u datoteku );
   halt(1).
end;
```

slika 9.6.

Novitet je komanda **seek** kojom se unutrašnji pokazivač na zadato mesto u datoteci – svako sledeće **read** će čitati podatke od tog mesta, dok će svako **write** upisivati tu. U primeru 9.6 morali smo dva puta da koristimo **seek** – jednom da bismo pročitali podatak a drugi put da bismo ga izmenili. Da je drugi **seek** izostavljen, ne bi program pogrešnosti slog (i mali bismo duplirana poruka o jednom računu, ali bismo ostali potpuno izgubili podatke o nekom drugom. Primetimo osim toga, da se uvek mora navesti broj sloga – nema mogućnosti da podatke čitanje od poslednjeg sakog sloga, jer bi podaci koje hsmo tako dobili svakako bili besmislem

Tipizirane datoteke se moraju se sastojati od komponenti definisanih slogova mogu da se koriste i skalarni tipovi kao što su **integer, real, extended**. Program se ali ke 9.7 učitava racionalne brojeve iz neke datoteke a zatim se na ekranu izdaje njihov zbir. Primetimo da je sa „sumiranje datoteke" korišćena procedura i da je čitava datoteka predmetom kao parametar– paskal zahteva da datoteka bude prosleđena po imenu (**var**) čak i ako njen sadržaj neće biti promenjen.

```
program sumafajla (output).

type  rfila = file of real;

var   f        rfile;
      fname    string.

procedure sumfile(var datafile
var   suma   real,
      k      real,
begin
{$I }
   reset (datafile),
   if ioresult<>0 then
   begin
      writeln( Datoteka ne
      exit
   end,
   suma :=0.
   while not eof (datafile) do
   begin
      read (datafile, x).
      if ioresult<>0 then
```

```
      begin
         writeln('Neu
datoteke');
         close(
         exit;
      end.
      sumn = sumn+x
   end,
   close (datafile).
   writeln ( Suma je:
end.

begin
   repeat
      write  ( ime datoteke:
      readln (fname)
      if fname =   then
      assign (f. fname);
      sumfile (f),
   until fname
end.
```

slika 9.7

Specijalni slučaj ovih datoteka bio bi **file *of char*** – tekstualna datoteka. Umesto ove deklaracije, shvatamo treba koristiti složenu **ret text** na primer **var izvestaj: ssim** Take nastaje tekstualna datoteka koja se dotuje meda čitati DOS komandom TYPE ili menjati primenom bilo kojeg tekst editora. Za čitanje podataka iz ove kve datoteka koristi se komanda **readln** a za upis **write** ili **writeln**, savisno od toga da li datoteci upisom treba zaključiti red ili ne

Program 9.8 ilustruje rad sa tekstualnim datotekama – rezlovi ulazne datoteke prepisuju se u izlazna pri čemu se iz svakog reda izbacuju svi kontrolni kodovi manji od 32 ili veći od 128

```
program izpsna.

var ulaz, izlaz  text,
    redu, redi   string,
    i  integer

begin
   assign (ulaz,   pasajni.txt');
   assign (izlaz,  ciljni.tx
{$i }
   reset (ulaz),
   if ioresult<>0 then
   begin
      writeln ( Ne mogu da otvorim datoteku
POCETNI.TXT ),
      halt (1).
   end.
   rewrite (izlaz).
   if ioresult<>0 then
   begin
      writeln ('Ne mogu da otvorim datoteku
CILJNI.TXT ):
      halt (1).
   end,
   while not eof(ulaz) do
   begin
      readln (ulaz, redu),
      if ioresult<>0 then
      begin
         writeln ( Neuspela čitanja ulazne
datoteke );
         halt (1),
      end,
```



```
      redi = .
      for i =1 to length (redu) do
         if (redu[i]>=chr(32)) and
du[i]<=chr(127)) then
            redi =redi+redu[i].
      writeln (izlaz, redi),
      if ioresult<>0 then
      begin
         writeln ( Neuspeo upis
         halt (1),
      end.
   end,
   close (ulaz).
   close (izlaz),
end
```

slika 9.8.

Tekstualne datoteke su mnogome karisne, pošto je rad sa njima razvan jednostavan a performanse koje se postižu zvanredne Zapamtite da **nikada** ne treba koristiti deklaraciju **file of byte** – sintaksno će sve biti ispravno, program će korektno raditi ali će biti tehko spor da će vas to iznenaditi Ukoliko dakle, možete da zamenite **file of byte** sa **text**, svakako te učinite nekada će, među tim tekstualna datoteke biti neupotrebljive jer se po njima ne može slobodna kretati komandom **seek** (ili bar ne može bez „prljavih trikova"). U takvim slučajevima treba pribeći netipiziranim datotekama

Netipizirane datoteke

Netipizirane (**untyped**) datoteka se deklarišu se var **ime file** – o obzirom da tip nije naveden *Turbo Pascal* na zna šta se nalazi u takvoj datoteci pa je tretira kao niz bajtova ili bolje rači niz blokova Jedan blok se iz nekih interijskih razloga sastoji od 128 bajta, ali često praktično uvek promenićli ovu konvenciju i raditi se blokovima dužine jedan bajt zato se netipizirana datoteka **uvek** otvaraju se **reset (ime, 1)** ili **rewrite (ime, 1)** – broj 1 je dužina bloke u bajtovima

Čitanje i pisanje netipiziranih datoteka povereno je komandama **blockread** i **blockwrite** – u pitanju je, zapravo „preskakivanje" odredenih DOS funkcija, tako da je rad izuzetno brz Upamtimo ga na jednostavnom primeru

```
program kopire;
var ulaz, izlaz  file,
    nuar, nuww  word,
    buf  array (1  2048] of char,
begin
   assign  (ulaz, parametr(1)),
```

```
   reset   (ulaz, 1),
   assign  (izlaz, parametr(2)),
   rewrite (izlaz,1).
   writeln ( Kopire se    filemize (niaz),
' de)ieve    ),
   repeat
      blockread  (ulaz.  buf, sizeof(buf),
Nuar)
      blockwrite (izlaz, buf, nuar, nuww).
      if nuww<>nupr then
      begin
         writeln ( Neuspeo upis na
         halt (1).
      end,
   until (nuar<>sizeof
   close (ulaz).
   close (izlaz).
end.
```

slika 9.9.

Program 9.9 zapravo kopira ulaznu datoteku u izlaznu pozv KOPIRA PRVI.DAT DRUGI.DAT će prekopirati PRVI.DAT u DRUGI.DAT Program najpre proverava parametre (**parametr(1)**) je prvi parametar navedan iza imena programa, **parametr(2)** je drugi) otvara datoteke i vrše puta poziva **blockread** kako bi sa podaci učitah. Prvi parametar procedura **blockread** je ime datoteka, drugi je lokacija bafera u koji se podaci upisuju (mi smo poprosto definisali niz od 2048 bajtova) treći je broj bajtova koje treba pročitati (u našem slučaju sizeof(**buf**) odnosno 2048), a četvrti vraća broj bajtova koji su stvarno pročitani. Obično se pročita onoliko koliko se i traži Ali, pri nailasku na kraj datoteke biće pročitano manje po ča program, pošto tpak upiše podatke na disk, isači iz petlje.

Procedura blockwrite una slične parametre – jedino novitet je što razlika broja željenih i stvarno upisanih bajtova signalizira neki ozbiljan problem, obično pun disk. Komanda **seek** se koristi za slobodno kretanje po netipiziranoj datoteci – ostavljamo vam da, na vežbu, sa nu napišete program koji će kopirati samo deo izabrane datoteke

Program 9.9 bi se mogao znatno ubrzati korišćenjem većeg bafera, ali za toda numerično na ograničenjem segmenta podataka – ako za taj bafer predvidimo 60 K, možda nam neće ostati dovoljna mesta za ostale promenljive! Rešenje je u korišćenju dinamičkih podataka – kada pročitate sledeće poglavlje i prostudirate program 10.5 lako ćete povećati bafer do teoryske granice od 64 K (više ne može zbog dizajna samog MS DOS-a), pa tro šeći praktično nemalo prostora u segmentu podataka.

10. RAD SA LISTAMA

Pažljivi čitaoci četvrtog poglavlja svog uvoda svakako su primetili da se pri dimenzionisanju nizova uvek mora zadati konkretna vrednost – neće prći nekakve učita vanje nekog n a onda definisanje niza čiji indeksi idu od 1 do n Ova je u praksi dosta neprijatna, pošto bi značilo da za svi nizovi maraju definisati prema maksimalnoj dužini koju, pod bilo kakvim uslovima, mogu imati. Kada se na to doda ograničenje od 64 kilobajta za sve podatke koja će program koristiti problem upravljanja memorijom postaje i te kako akutan– Razmotrimo dakle, mogućnost korišćenja dinamičkih struktura podataka – liste

Elementi liste zauzimaju minimum neophodnog prostora kada nam zatreba novi element, rezervišemo prostor za njega a čim nam taj element ne bude potreban–prostor oslobađamo za druge stvari. Dinamičke strukture ipak, nisu savršenstvo – za njima se neke operacije pago na matricama, a pogotove će teško biti onima koji se po prvi put sreću sa ukazateljima i sličnim dragunjama. I pak, vrna je srne.

Uvodimo najpre novi tip podataka – pointere ili ukazatelje Da bismo uklonili zabunu koju često izaziva činjenica da se promenljive tipa pointer takođe zovu pointeri, odlučili smo se da reč **pointer** koristimo kada govorimo o **tipu**, a njen prevod (**ukazatelj**) kada govorimo o **promenljivoj** Ukazatelj možete da zamislite kao neku vrstu adrese memorijske ćelije u kojoj je neki podatak upisan

Svaka promenljiva u paskalu, dinamička ili ne, ima svoj tip koji se obično definiše u sekciji **var** sekcije programa Liste na mogu ovako da se deklarišu, jer nemaju ime i jer je njihova veličina neodredena. Zato za sve di-

zamućkih struktura daje u okviru **type** sekcije, kao na slici 10.1

```
program lista (input, output);

type art_ptr = ^artikl;
     artikl = record
                sled  : art_ptr;
                ime   : string[20];
                cena  : longint;
              end;

var pocetni  art_ptr;
    kalje    sedlean;
    izker    integer;
```

slika 10.1

Deklaracije sa slike 10.1 uvele jednu listu čiji će elementi biti podaci o artiklima. ime i cena. Osim ova dva broja svaki slog mora da sadrži i ukazatelj na sledeći slog u ulančanoj listi. Prvi slog će, dakle ukazivati na drugi, drugi na treći i tako dalje, poslednji će slog ukazivati na „ništa", pri čemu se ta „ništa" za pascala zove **nil**. Šta nam nedostaje? Naravno, nekte treba da ukazuje na prvi slog, pa smo uveli promenljivu **pocetni** tipa pokazivač.

Da li ste primetili da slika 10.1 prikazuje nekte veoma neobično se pascal tip **art_ptr** se definiše kao „**artikl** pri čemu će tip **artikl** biti definisan tek u sledećem redu – da nada zna emeh da koristimo takvučve strukture koje smo već Pefinisali! Ovo se odstupanje od dobrih pravila strukturiranog programiranja, za žalost, nije mogle izbeći. Da smo najpre definisali tip **artikl**, za bi smo mogli da opišemo jedno od njegovih polja koje je o-bavezno pointer tipa **art_ptr**!

```
procedure nbaci(var pocetni  art_ptr;
                    ime  string;
                var lenpint);

begin
  if pocetni =
  begin
    new (
    pocetni^.ime :=
    pocetni^.
    pocetni^.
  end
  else nbaci(p
end;
```

slika 10.2.

Na slici 10.2 vidimo proceduru koja dodaje elemente listi. Proceduri je, jasno, potrebna inicijalizacija. u glavnom programu treba da napišemo **pocetni:=nil** što znači da je lista prazna. Kada pešalimo da joj Pašamo neki element, koristićemo ugrađenu proceduru **new** pozivom **new(pocetni)**: računar će rezervisati prostor se prvi element liste, a zatim ukazatelju **pocetni** Podešiti adresu tog slobodnog prostora. Sada možemo da Podešljivo vrednosti **pocetni^.ime** na primer, osnovnu polje ime u zapisu na koga pokazuje **pocetni**.

Kako radi procedura se slike 10.2? Kao što primećujete rekurzivna je – pretražuje redom elemente liste dok ne nade poslednji (se nil ukazatelјem). zatim kreira novi i ulančava ga. Vaspne je bitno da je ulazni ukazatelj na početku prenosa pa snemu (se var) – prekjete šta h se desilo ako bi se ta tri slova izostavila.

Pošto smo primenom procedure nbaci napunili listu, pošalesdeno da je čitamo. U tome nam pomaže procedura ispis, koja spisuje elemente liste Analiziraje je i sa-

kljnčite zašto se ulazna veličina **nocen** prenosi pa vrednosti a ne pa imenu!

```
procedure print(pocetni  art_ptr);
begin
  while pocetni<>nil Po
  begin
    writeln('Artikl   , pocetni^ ime)
    writeln( Cena     , pocetni  cena),
    writeln,
    pocetni =pocetni^ sled
  end
end;
```

slika 10.3

Procedura **brisi** na slike 10.4 omogućava brisanje zapamćenih elemenata liste Treba zapaze prvenci odgovarajući element, a zatim modifikovati ukazatelj prethodnog elementa tako da pokazuje na sledeći Taku ći element, naravno nije „mrtav" premda oko se pokazuje na njega treba se poslužiti procedurom dispose(a) koja oslobađa prostor na kaje je X ukazivao

```
procedure brisi(var pocetni  art_ptr, ilsa.
string),
var prethodni, tekuci  art_ptr,
    Polje  Seolean,
begin
  tekuci =pocetni, srethePni =nil,
  while (tekuci^ ime<>iime) and
(tekuci  sled<>nil) do
    begin
      prethodni =tekuci,
      tekuci =tekuci^ sleP,
    end,
  if tekuci  ime=iima then
  begin
    if srethedni=nil then
      pocetni :=tekuci^ sleP
    else
      prethedni^ sled:=tekuci^.
    Pispose(tekuci)
  end
  else writeln ('Element nije
end,
```

slika 10.4

Ovaka izležene lista su element kako standardnog pa skala tako i *Borland*-ovog kompajlera. U raznim knjigama pročitaćete mnogo lepih reči o njima, kao i o problemima koja se uz pomoć njih laks rešavaju. Nevolja sa takvim problemima (sređićete ih sa svakom novom iz pascala, pošto se smatra da su liste „najzanesljenji" deo svega jezika) je što su „izmišljeni" tako da se rešavaju uz pomoć lista U praksi takvi su problemi veoma retki (ako ih uopšte ima) a ručno strana lista redovno Pelazi da urađaje elementima se mnogo pristupati sekvencijalno tj sa preskočenje hijadjeg elemenle mora se pročitati prvih 999' Zbog toga je čitave ova priča se **new/dispose** jedva vredna vaše pažnje. Pe ipak, dinamičke strukture se u *Turbo Pascal*-u neophodna pre svega zbog ograničenja maksprocesora 80x86 na kojem su zasnovani današnji personalni računari

640 kilobajta sistemerje vašeg PC-ja podaljena je na oskolko sekina. Tu je, pre svega, prostor koji zauzime DOS i eventualni rezidentni programi Zatim sam pascal program koji se uvrštava i njegovi podaci koji videli ono, mogu da zauzmu do 64 kilobajta Ostatak memori je da 640 K (ili više, ako koristite QEMM ili neki sličan program) je slobodan – nazivamo ga **heap** i koristima se dinamičke strukture podataka.

Pogledajmo primer 10.6 u kojem se prostor na **heap**-u koristi za smeštanje niza Pugeg geteve 64 kilobajta. Osaovnih 64 K za podatke je „ostakauto" što znači da se nekakiko ovaka Deklaracionik nizova možemo prevazići jedno od najvažnijih ograničenje *Turbo Pascal*-a.

```
Program heap;

const maxatr = 64000,

type dugistr     record
                   Pnzina  longint,
                   txt  array [1  maxatr]
char,
                 enP
     ukazt       dugistr,

var ukez  nkazt,

procedure Podeli (ukaz
longint],
var x  lenpint,
begin
  nkaz^ duzine =0,
  for i:=1 to kolike do
  begin
    ukez  txt[i].=chr (ranaom (od|*32);
    ine(nkaz^ Puzine),
  end,
enP,

procedure ispisi (ukez  ukazt),
```

```
var if longint;
begin
  for i:=1 to nkaz^ Puzine Po
    write (nkaz^ txt[i]);
  writein,
end,

begin
  if mekevail<sizeof(dugistr)
  begin
    writein ( Neuspela alokacija
    rije ),
    halt(1),
  end,
  gatman [ukaz, sizeof (Pugistr
  dodeli [ukaz, 64000],
  ispisi [ukaz],
end
```

slika 10.5.

Ukolika rešite sa bile kajom verzijom *Turbo Pascal*-a, na ovaj način možete da alocirate tačno onoliko memori je kolko imate – čim je nestvoru RAM pun, alokacija neće uspeti, čak i ako vaš PC ima 7 megabajta slobodna memorije usred 640 teg kilobajta. Kod *Borland Pascal*-a 7.0 sve je drugačije – možete raspolagati kompletnom memorijom (Pa 16 megabajta) pa čak i pod određenim uslovima višak podataka automatski odlagati na disk. Na žalost i dalje je prisutno ograničenje od 64 kilobajta po strukturi – možete da alocirate nanoge struktura, ali i dalje nšta od jednog xima kaja bi bio dug 5 megabajta!

11. TURBO PASCAL I DOS

Paskal, kao što znamo iz senovi nije namenjen pisanju sistemskih programe ali ne *Borland*-ovi programeri potrudili da *Turbo Pascal* bude spremen i za takva iku šenje – uvedene su brojne ekstenzije koje su učinile da mnogi posneći sistemski i uslužni programi za PC računare budu napisani baš na paskalu Jezik najbolje prilagođen pisanju sistemskih programa je naravno C

Pristup memoriji i periferiji

Prva stvar koja je neophodna za pisanje sistemskih programa je mogućnost slobodnog čeprkanja po memoriji, tj. oslobađanje ograničenja koja nameću PEEK i POKE Konstruktori *Turbo Pascal*-a su verovatno mogli da im plementiraju istenzane procedure ali su se setili sebeg bolyeg – definisan je **seoPo** nis **meom** čijim se elementi ma pristupa sa **meom(segment.ofset)**, gde su **segment** i **ofset** segmentna odnosno ofsetna adresa bajta kojem se pristupa ako su in^aka promenljive, one traba da budu tipa **word**. Poeskad je pogodne pristupati računa otkazane dugim računa i u tom slučaju se koriste odgovarajući pseudo nazivi **meomw i meomf** Ne primer

```
meom[$0040 $004B] = 7,
reč                = eomw[seg(v) ofs(v)];
Puge_reč           = eomf[64 3*4],
```

slika 11 1

Hardverski orjentisani programi treba a vremena za vreme da pristupaju i pertavima. Primerjena je naevim sličnu metodologiju pseudo nazivi **port i portw**

```
portw[$20] =$2020,
port[baza] =port[baze] xor aeska,
while port[$62] enP $80 = 0 Po ..
```

slika 11.2.

Pisanje interapt procedura je nešto složenije Oblik svake interapt procedure pre svega treba da buda

```
procedure ime (flags,ee,ip,ax,bx,cx,
                          dx,si Pi,Pa,as,bp,
Interrnet
Cegin
  .
end;
```

slika 11.3.

Interapt procedura automatski pobacuje sadržaj svih registara na stak i inicijalizuje DS registar što se svodi na

```
PUSH AX
PUSH BX
PUSH CX
PUSH DX
PUSH SI
PUSH DI
PUSH ES
PUSH BP
MOV  BP,SP
SUB  BP,LocalSize
MOV  AX,SEG DATA
MOV  DS,AX
```

slika 11 4

Oštta osobnjaje instrukcija CLI kaja onemogućava interapte dodatnene je na inline ako smatrame da je potrebna. I pored čuvanje na staku interapt procedur je dopnitneme da menje vrednosti deklarisanih parameta-

ra, take da prenese „prečve" kraj izvršavanja, pre nego što se odluče za prenesu svakog parametra treba, naravno da dobro znate šta radite. Interapt procedura koja skraćuje hardverski generisanje prekida treba da se koristi svih *Turbo Pascal*-ovih I/O procedura ili procedura sa sleceranje memorije potre one nisu reesntrant. Iz istog razloga se smaju da se koriste DOS funkcije

Štampanje iz paskala

Mnogima interapt procedure i direktan pristup memoriji nisu potrebni: ali će svako poželeti nešto da odštampa. Najjednostavniji način da to uradite je da tekstualnoj datoteci dodelite fizičke ime LPT1 (ili LPT2 LPT3 zavisno gde je priključen štampač) otvorite je (sa rewrite) i onda upisujete podatke u nju. Ta metodologija dobro funkcioniše dok se štampa tekst ali kod grafike ili definisanih karaktera može vas iznenaditi sasvim pogrešan ispis. Problem leže u *Turbo Pascal* u koje u DOS-u kojem se paskal obraća za „usluginu" iz interapta raziga, DOS smatra kod chr(26) za kraj datoteke i na prsleduje ga štampaču. Rešenje je u procedure **dos_rowmode** koju treba pozvati pe otvaranju datoteka ke u primeru 11 8

```
program stampa,

uses dos,

var stampac  text,
    i  integer,
```

```
procedure dos_rowmode (var datot)
var rge  registers
    fh   integer,
    fptr  ^integer
begin
   fptr :=ptr(seg(datot)  ofs(datot))
   fh :=fptr^
   rgs.ax :=$4400,
   rgs.bx  fh
   msdos (rgs)
   rgs.dx :=rgs.dx and $ff
   rgs.dx :=rgs.dx or 32
   rgs.ax :=$4401
   rgs.bx :=fh,
   msdos (rge),
end,

begin
   assign (stampac   lpt1 )
   rewrite (stampac)
   dos_rowmode (stampac)
   for i :=32 to 254 do
       write (stampac  chr(i))
   writeln (stampac),
   close (stampac),
end
```

slika 11.5.

DODATAK: MALA BIBLIOTEKA TURBO PASCAL FUNKCIJA

U okviru svog poglavlja ukratko ćemo opisati razne izgrađene procedura i funkcije *Turbo Pascal*-a – naročiti zato ona koja su najpotrebnije za pisanje uslužnih i sistemskih programa. Pri opisu parametara svaka od funkcija koristeće su sledeće konvencije

by	byte
wo	word
sh	shortint
in	longint
i	integer
ch	bilo koji od prethodnih tipa, wo, sh, in, i
r	real
bo	bilo koji od tipova sh/r
bo	Boolean
ch	char
st	string
tx	tekstualna datoteka
by	datoteka određenog tipa
ut	netipizirana datoteka
f	bilo koja datoteka

Rad sa stringovima

■ **function Concat (st1 st2, , , stN)· st**

Konkatenacija, odnosno nadovezivanje stringova **st1**, **st2** – , **stn** i vraćanje rezultujućeg stringa Ova funkcija je u izvesnom smislu jedinstvena, pošto ima promenljiv broj argumenata

■ **function Copy (st), i1, i2): st**

Izdvajanje podstringa koji ima **i2** znakove počevši od znaka čija je redni broj u stringu **st1** jednak **i1** Ukoliko **i1** pokazuje na kraje **st1** funkcija vraća prazan string Ukoliko se izborom veličog **i2** zahteva kopiranje slova koja se postoje funkcija će vratiti segment stringa **st1** od **i1** sg znaka do kraja. Primer a:='Pascal', b:=copy (a,3,2) daje vrednost 'sc'

■ **procedure Delete (st, i1, i2)**

Uklanjanje **i2** znakova u stringa **st** i ta počevši od znaka čije je poziciju **i1** Ukoliko **i1** pokazuje ima kraje **st** funkcija neće uklentu ni jedan znak. Ukoliko se izborom velikog **i2** zahteva brisanje slove koja ne postoje, funkcija će bi vratiti segment stringa **st** od početka do znaka (**i1** 1) Primer a:='Pascal', delete (a,3,2) daje vrednost 'Pasl'

■ **procedure Insert (st1, st2, i)**

Umetanje stringa **st1** u string **st2** i to počevši od pozicije **i** Ukoliko je rezultujući string duži od 255 znakova, višak će biti odsečen. Primer a:='Pascal'; insert (' l', a, 4) daje 'Pasl sal

■ **function Length (st)· i**

Vraća dužinu stringa **st**. Ako je string prazan, dužina je nula. Primer a:='Pascal'; i:=length (a) daje vred nost 5

■ **function Pos (st1, st2) by**

Pronalazi prvo pojavljivanje stringa **st1** u stringu **st2** i vraća njegovu poziciju ili nulu ako **st1** nije pronađen. Primer a:='Pascal', b =pos ('sc' a) daje vrednost 3

■ **procedure Str (ir, st)**

Konverzija bilo kog cel ili racionalan broj **ir** u string **st** Konverzijom se može upravljati kao i u naredbi **write str (25:4, a)** daje string ' 25 Konverzijom racionalnih brojeva se upravlja se **i:razn:strima.decimala.**

■ **function OpCase (ch)· ch**

Pravedi karakter **ch** u velika slove. Ako ch nije mala slove ostaće nepromenjena Ne treba ni spominjati da **UpCase** ne podržava YU slova.

■ **procedure Val (st, ir, i)**

Konverzija stringa **st** u cel ili racionalan broj **ir** Ceo broj i opisuje uspeh konverzija – i=0 označave da je konverzija uspela dok će ako konverzija ne uspe i vratiti poziciju prvog znaka koji je pravio probleme

Rad sa datotekama

■ **procedure Assign (f, st)**

Logičke ime **f** se povezuje sa datotekom na disku ot U kolika je st prazan string **f** će ukuduće biti tastatura (ekran odnosno tastatura) ali na specijalan način koji podržava redirekcije MS DOS-a.

■ **procedure ChDir (st)**

Menja radu direktorijum u ot Vaši MS DOS alutskas

■ **procedure Close (f)**

Zatvaranje datoteke Ukolike su u datoteku upisivani neki podaci se close zahtevame da se kompletan sadržaj bafera koji je DOS koristia prenese na disk Datoteka treba obavezno zatvarati, jer neregularan kraj izvršava nja programa može da dovede do gubitke dela podataka u nezatvorenim datotekama.

■ **function eof (f), ba**

Vraća **true** ako je markac pozicioniran na kraj datoteke **f** tj ako je dalje čitanje besprednetno

■ **procedure Erase (f)**

Brisanje datoteke **f** koje ne sme biti otvorene

■ **procedure GetDir (by, st)**

Preuzanjiva **st** dužina sadržaj koji je jednak punom imenu radnog direktorijuma **By** je tiesni parametar koji opisuje disk – 0 je radni disk 1 je A 2 B 3 C i tako dalje

■ **function IOResult wo**

Funkcija ima smisle samo ako je se {$I i isključene automatske prijavljivanje I/O greški Tada posle svake operacije možemo da izvršime **loresult** i tako dstektujemo grešku – nula označava uspešnu operaciju, a broj različit od nule neku od greški

■ **procedure MkDir (st)**

Kreiranje direktorijuma čije je ime zadato sa st Pri zadavanju imena treba koristiti standardnu sintaksu MS DOS-a.

■ **procedure Rename (f, st)**

Menja ime datoteke **f** u **st** Datoteka na sme da bude otvorena – redosled stvari je **assign ...-rename** ili **assign —reset/rewrite –. close –. rename.**

■ **procedure Reset (f, wo)**

Otvara datoteku **f** za čitanja i pemcionra pointer na njoe početak Parameter **wo** je opcioni i dopušten samo kod netipiziranih datoteka pomoću njega određujemo veličinu bloke u bajtovima (podrazumeva se 128 ali se praktično uvek stavlja 1)

Sadržaj datoteke otvorene za čitanje može se i modifikovati

■ **procedure Rewrite (f, wo)**

Otvara datoteku **f** se upis unišiava njen eventualni ranij sadržaj i pemcionra pointer na sam početak. Parametar **wo** je opcioni i dopušten samo kod netipiziranih datoteka, pomoću njega određujemo veličinu bloka u bajtovima (podrazumeva se 128 ali se praktične uvek stavlja 1)

■ **procedure RmDir (st)**

Uklanjanje direktorijuma čije je ime (st) zadato u okiu du se ostanksen MS DOS-a Da bi direktorijum mogao da se obriše, u njemu se ne sme nalaziti ni jedna datoteka.

Rad sa tekstualnim datotekama

■ **procedure Append (tx)**

Otvara tekstualnu datoteku **tx** i priprema se za dodavanje podataka na njen kraj Ukolike je logičkom imenu dodeljen prazan string (npr **assign (tx, '')**) pristupade sa standardnoj izlaznoj jedinici tj ekranu monitora.

■ **function eoln (tx) bo**

Vraća **true** ako su svi znaci iz tekuće linije proč tir Tede treba primeniti **readln**

■ **procedure Flush (tx)**

Primenom ove procedura obezbeđuje se da kompletan sadržaj bafere koji je DOS otvorio kako bi komunicirao sa datotekom tx bude prepisan na disk, što znači da su u slučaju neregularnog završetka programa neće doći do gubitka podataka.

■ **procedure Read (tx, v1, v2, vn)**

Učitava i konvertuje podatke iz tekstualne datoteke **tx**. Svaki podatak mora biti kodiran u ASCII obliku i te u skladu sa tipom promenljive (**integer, real, Boolean, char ili string**) Ako se dostigne kraj reda, procedura vraća CR (chr(13)), ako se dostigne kraj datoteka, vraća Ctrl Z (chr(26))

■ **procedure Readln (tx, v1, v2, , n)**

Učitava i konvertuje podatke iz tekstualne datoteke **tx** kao i **read**. Razlika je jedino u tome što se po završenom čitanju preskače **eoln** marker i prelazi u sledeći red.

■ **function SeekEof (tx)· bo**

Vraća **eof** status tekstualne datoteke. Funkcija je identične se **eof** ali preskače eventualne tabove, blankove i prazne linije na kraju datoteke.

■ **function SeekEoln (tx), bo**

Vraca **eoln** status tekstualne datoteke Funkcija je identična se **eoln** ali preskače eventualne tabove i blanke ononle ne kraju reda

■ **procedure SetTextBuf (tx, buf, wo)**

Svakoj tekstualnoj datoteci interne se dodeljuje bafer od 128 bajta kojem pristupaju procedure **read** i **write** Za neke aplikacije veličina ovog bafera može da bude ograničenje pa je pradvidena mogućnost njegovog proširenje ili skraćenja pasle ove naredba datoteka **tx** će koristiti bafer **buf** (obično je **buf** neki niz čija je dimenzija **wo** Procedura **entiextbuf** obavezna treba izvršiti pre otvaranje datoteke tj izmedu **assign** i **reset/rewrite**

■ **procedure Write (tx, v1, v2, ... vn)**

Konvertuje podatke v1, v2, .. vn u ASCII oblik i upisuje ih u datoteku tx. Svi podaci moraju da budu tipa **integer**, **real**, **Boolean**, **char** ili **string** i njihovom se formatom može opravljati sa **specijalnecimala**. Write (tx, 1.0) na primer upisuje u datoteku tx vrednost promenljive 1 koju prethodno tačno pet mesta; ako je broj manji od 10000, dopunjava se vodećim blankovima.

■ **procedure WriteLn (tx, v1, v2, .. vn)**

Konvertuje podatke v1, v2, .. vn u ASCII oblik i upisuje ih u datoteku tx. Razlika u odnosu na **write** je što se posle završenog upisivanja dodaje i **eoln** marker tj. prelazi u sledeći red.

Rad sa netekstualnim datotekama

■ **function FilePos (f): li**

Vraća trenutnu poziciju pointera datoteke f u vidu long-int-a. Ako je pointer na početku datoteke, funkcija vraća nulu.

■ **function FileSize (f): li**

Vraća veličinu datoteke f u označenim jedinicama ako je datoteke netipizirana, označene jedinice su blokovi. Broj je veličina sadata u okviru procedure koja je otvorila datoteku ako je datoteka tipizirana, veličina je broj slogova.

■ **procedure Read (ty, v1, v2, ... vn)**

Učitavanje podataka v1, v2, .. vn iz datoteke ty. Promenljive moraju da budu usaglašene sa tipom datoteke obično se radi o slogovima. Ukoliko je pre čitanja uslog podatka **eof** bilo **true** nastupa greška

■ **procedure Seek (f, li)**

Pozicionira pointer na li-ti element datoteke f. Smatra se da je prvi element (npr. slog) datoteke obeležen brojem 0. Pointer se pozicionira sa sam kraj datoteke (na poslednjeg elementa) sa **seek (f, filesize (f))**

■ **procedure Truncate (f)**

Ovu proceduru izvršavamo kada želimo da uništimo podatke u datoteci i to od tekuće pozicije pointera pa do njenog kraja. Jednom uništeni podaci obično se ne mogu povratiti.

■ **procedure Write (ty, v1, v2, ... vn)**

Upis podataka v1, v2, .. vn u datoteku ty. Promenljive moraju da budu usaglašene sa tipom datoteke obično se radi o slogovima. Pri primeni procedure **write** na datoteke koje nisu tekstualne upravljanje izpisom se **širina.decimala** nema smisla, pa prema tome nije ni omogućeno

Rad sa netipiziranim datotekama

■ **procedure BlockRead (f, buf wo1, wo2)**

Učitava jedan ili više blokove podataka iz netipiziranog datoteke f u bafer **buf**, veličine bloke je definisana u proceduri koja je otvorila datoteku (**reset**), pri čemu je podrazumevana vrednost 128. **Wo1** određuje koliko će se blokova kopirati. **wo2** daje izveštaj o čitanju: ukoliko je **wo2 = wo1** učitani su svi traženi blokovi ako je **wo2<wo1** učitano je samo **wo2** blokova a onda je dostignut kraj datoteke

Buf je bilo koja struktura podataka, na primer promenljiva ili (delić) niz. Podaci će se upisivati u memorijski prostor koji zauzima promenljiva **buf** ako tamo nema dovoljno mesta, snimiće se onoliko da će biti uništeni neki drugi podaci što obično izaziva krah sistema

■ **procedure BlockWrite (f, buf wo1, wo2)**

Upisuje jedan ili više blokove podataka iz memorijskog bafera **buf** u netipiziranu datoteku f veličina bloka je definisana u proceduri koja je otvorila datoteku (**rewrite**) pri čemu je podrazumevana vrednost 128. **Wo1** određuje koliko će se blokova kopirati. **wo2** daje izveštaj o izvršenoj operaciji – ukoliko je **wo2 = wo1**, upisani su svi potrebni blokovi dok **wo2<wo1** signalizira da je došlo do upisivanja **wo2** blokova, dakle do I/O greške, koja se obično svodi na prepun disk.

Sistemske procedure

■ **function Addr (x): pointer**

Vraća pointer na objekt **x**, baš kao i operator @

■ **procedure ClrScr**

Briše sadržaj ekrana.

■ **function CSeg: wo**

Vraća trenutni sadržaj registra CS

■ **procedure Dec (x, n)**

Umanjuje skalarnu promenljivu bilo kog tipa za n (ili, ako se ne zadaje, za jedan). Izvršava se brže nego **x:=x-1** pa je pogodna za sistemske programe

■ **procedure Delay (wo)**

Čeka wo milisekundi.

■ **function DiskFree (wo): li**

Vraća broj slobodnih bajtova na drajvu wo. Nula označava tekući drajv, jedan drajv A, itd.

■ **function DiskSize (wo): li**

Vraća ukupan broj bajtova na drajvu wo. Nula označava tekući drajv, jedan drajv A, itd

■ **function DosExitCode: wo**

Vraća izlazni kod podprocesa. Nula označava normalno završetak, jedinica prekid sa Ctrl Break, dvojka prekid sleep, neraspoloživog uređaja, a trojka da je pozvani program ostao u memoriji kao rezidentan

■ **function DosVersion: wo**

Vraća aktivnu verziju MS DOS-a. Niži bajt rezultata je uvek jedinica, pošto vrsta označu osnovna revizije (2, 3 ili 4) viši bajt vraća broj koji se pri normalnom ispisu nalazi sa desne strane tačke. Na DOS-u 3.30, na primer, viši bajt će imati vrednost 30 a niži vrednost 3

■ **function DSeg: wo**

Vraća trenutni sadržaj registra DS

■ **procedure Exec (st1, st2)**

Izvršava program koji se zove **st1**, preuzimajući mu parametar **st2**. Ako je u pitanju .EXE ili .COM program more se navesti njegovo puno ime, koje uključuje kako stazu (**path**) tako i ekstenziju. Ako želite da izvršite neku DOS komandu ili .BAT fajl uradite to posredno preko kopije COMMAND.COM-a. Na primer **exec ('c:\command.com', '/C dir *.pas')**; Ne zaboravite da se ovakve „emitovane procese“ treba eliminisati nekim

[illegible] unjavljuje programom se na primer ISM [illegible]

■ **function FExpand (st1): st2**

Kompletira [illegible] u puno ime koje uključuje [illegible] direktorijum i ceste datoteku. Prilikom ekspanzi[illegible], te imene se uklanjaju i reference [illegible] u [illegible]. Ukoliko je ne [illegible] na primer, tekući [illegible] [illegible] [illegible]